CBR SYSTEM FOR IDENTIFYING TRADITIONAL CHINESE HERBS

SESSION 2002/2003
WXES 3182 PROJEK ILMIAH TAHAP II

CHAN HO SANG
WEK 000124

SUPERVISOR : ASSOC. PROF. DR. SYED MALEK FAKAR DUANI
MODERATOR : EN WOO CHAW SENG
DATE OF SUBMISSION: 13 FEBRUARY 2003

ABSTRACT

This project documentation is an exercise submitted to the Faculty Computer Science and Information Technology, University of Malaya, as a first stage report for the Final Year Project, WXES3181 which in turn serves as a partial fulfillment for the Bachelor of Computer Science degree.

This project aims to demonstrate the feasibility of the application of the Case-Based Reasoning (CBR) method in identifying Traditional Chinese Herbs for certain illness or diseases. CBR has been proposed for tasks in which past experiences are exploited and adapted to solve the current problems. Art *Enterprise is the CBR development tool used.

In the beginning, the system will be introduced and its objectives will be defined together with the project schedule. The following chapter will present some explanations and background of AI and CBR technologies; Traditional Chinese Medicine and Herbs; and development tools used in implementing the system. Then, the methodologies will be discussed and reasons for the choice of case-based reasoning instead of other reasoning techniques are explained. The development tools and system requirements are also outlined. Finally, the result of system design will be presented in the final chapter of the proposal. The Data Flow Diagram and user interface design were outline in this chapter too.

Acknowledgement

First of all, I would like to greet all my readers a warm welcome. There are a few people I would personally like to take this opportunity to thank them.

I would like to express my sincerest gratitude and appreciation to my project supervisor Assoc. Prof. Dr. Syed Malek Fakar Duani for his invaluable advices and guidance to me throughout the whole project development.

Special thanks also go to my project moderator, Mr. Woo Chaw Seng for his guidance and advices in improving the system through various ways.

Last but not least is a special thanks to my family, friends, house-mates and classmates for their support and assistance when I needed them most in contributions to the success and completion of the project.

DISCLAIMER

TABLE OF CONTENTS

ABSTRACT I
ACKNOWLEDGEMENT II
DISCLAIMER III
TABLE OF CONTENTS IV
LIST OF FIGURES VII
LIST OF TABLES VIII

CHAPTER 1: INTRODUCTION
1.1 GENERAL VIEW 1
1.2 DEFINITION 1
1.3 OBJECTIVES 4
1.4 SCOPE 6
1.5 PROJECT ASSUMPTION 7
1.6 MOTIVATION FACTOR 7
1.7 SYSTEM LIMITATION 8
1.8 SCHEDULE 9
1.9 REPORT LAYOUT 10
1.10 SUMMARY 12

CHAPTER 2: LITERATURE REVIEW
2.1 INFORMATION SOURCE AND FINDING 13
2.1.1 Internet 14
2.1.2 Articles from Medias 14
2.1.3 Group Discussions 15
2.1.4 Others 15
2.2 ARTIFICIAL INTELLIGENCE 16
2.2.1 Introduction to Artificial Intelligence 17
2.2.2 The History of Artificial Intelligence 21
2.3 CASE-BASED REASONING 22
2.3.1 History of CBR 22
2.3.2 Description of CBR 25
2.3.3 How CBR Work? 27
2.3.4 CBR and Other Techniques 40
2.3.5 CBR Applications 41
2.3.5.1 Classification of CBR Applications 42
2.3.5.2 Implementation of Academic Research 43
2.3.6 Advantages of CBR 47
2.3.7 Disadvantages of CBR 48
2.3.8 CBR and Other Reasoning Method 48

2.4 TRADITIONAL CHINESE MEDICINE 49
2.4.1 Chinese Medicine and Western Medicine 50
2.4.2 Traditional Chinese Herbs 51
2.4.2.1 Introduction to Traditional Chinese Herbs *52*
2.4.2.2 Conditions not appropriate *55*
2.4.2.3 Herbs combining to make an effective treatment *55*
2.5 CASE-BASED REASONING DEVELOPMENT TOOLS 57
2.5.1 A Theoretically Ideal Feature Set 57
2.5.2 CBR Software Tools 59
2.6 REVIEW OF EXISTING COMMERCIAL SYSTEM 63
2.6.1 CaseLine 53
2.6.2 CLAVIER 55

CHAPTER 3: METHODOLOGY & SYSTEM ANALYSIS
3.1 CONCEPT OF METHODOLOGY 68
3.1.1 Waterfall Model 68
3.2 CASE-BASED REASONING OVERVIEW 72
3.2.1 Case Base Building and Usage 73
3.2 CBR VS OTHER REASONING TECHNIQUE 75
3.2.1 CBR Vs Rule-Based Reasoning 75
3.2.2 CBR Vs Model-Based Reasoning 76
3.2.3 CBR Advantages over Other Reasoning Techniques 77
3.3 DEVELOPMENT TOOLS 78
3.3.1 Hardware Tools 78
3.3.2 Software Tools 78
3.3.2.1 Art Enterprise* *79*
3.3.2.2 Windows 2000 Professional *79*
3.3.3 System Recommendation 80
3.4 SYSTEM REQUIREMENT 81
3.4.1 Functional Requirement 81
3.4.2 Non-Functional Requirement 82

CHAPTER 4: SYSTEM DESIGN
4.1 SYSTEM DESIGN OVERVIEW 83
4.2 SYSTEM ARCHITECTURE 85
4.3 DATA FLOW DIAGRAM (DFD) 86
4.3.1 Identify Module 89
4.3.2 Match Module 90
4.3.3 Reuse Module 91
4.3.4 Adaptation Module 92
4.3.5 Evaluation Module 93
4.3.6 Retain Module 94
4.4 CASE BASE DESIGN 94
4.5 SYSTEM USER INTERFACE 98

CHAPTER 5: SYSTEM IMPLEMENTATION
5.1 DEVELOPMENT ENVIROMENT 100

5.1.1 Hardware Tools 100
5.1.2 Software Tools 100
5.2 SYSTEM IMPLEMENTATION 100
5.3 FUNCTIONS IMPLEMENTATION 101
5.4 ART SCRIPT IMPLEMENTATION 106
5.5 WORD PREPROCESSING IMPLEMENTATION 110
5.5.1 Lexicons of Spell Checker 110
5.5.2 Word Preprocessing 111
5.6 AHP WEIGHTED k-NN ALGORITHM IMPLEMENTATION 114
5.6.1 Traditional k-nearest neighbor algorithm 114
5.6.2 AHP approach to k-NN algorithm 115
5.6.3 Advantage of AHP approach 116
5.6.4 AHP Modeling for Traditional Chinese Herb recommendation 116
5.6.5 Indexing with AHP feature weight and weight value assign 118
5.6.6 Case retrieving with Inductive Retrieval algorithm 119
5.6.7 Similarity of Case Matching 122
5.6.8 Comparison between k-NN algorithm and AHP weighted k-NN algorithm 124

CHAPTER 6: SYSTEM TESTING
6.1 INTRODUCTION 126
6.2 MODULE TESTING 127
6.2.1 Case Matching Accuracy Testing 127
6.2.2 Weighting Module Testing 127
6.2.3 Case Base Module Testing 128
6.2.4 Coding Module Testing 129
6.3 INTEGRATION TESTING 130
6.4 SYSTEM TESTING 130
6.4.1 Rule Testing 131
6.4.2 Performance Testing 131
6.5 TESTING ANALYSIS 131

CHAPTER 7: SYSTEM EVALUATION & CONCLUSION
7.1 SYSTEM EVALUATION 132
7.2 PROBLEMS ENCOUNTERED AND SOLUTIONS 132
7.2.1 Difficulties in choosing development technologies and tools 132
7.2.2 Lack of knowledge of Traditional Chinese Herbs 133
7.2.3 Lack of knowledge of Case-Based Reasoning 133
7.2.4 Lack of knowledge in choosing and implement an appropriate algorithm for case retrieval and indexing 133
7.2.5 Lack of knowledge in Art Script 134
7.3 SYSTEM STRENGTHS 135
7.4 SYSTEM LIMITATIONS 136
7.5 PROBLEM IN IMPLEMENTATION 137
7.6 FUTURE ENHANCEMENT 137

BIBLIOGRAPHY 139
APPENDIXES 142

LIST OF FIGURES

PAGE	FIGURE
21	Figure 2.1: Timeline of major AI event
27	Figure2.2: CBR Cycle (Aamodt & plaza, 1994, AI Communication)
39	Figure 2.3: A Task-method Decomposition of CBR
43	Figure 2.4: A Classification Hierarchy of CBR Applications [Althoff et al., 1995]
69	Figure 3.1: Waterfall Model
85	Figure 4.1: System Architecture
88	Figure 4.2 Context Diagram of CBR System
88	Figure 4.3: Data Flow Diagram of CBR System
89	Figure4.4: Identify Module
90	Figure4.5: Match Module
91	Figure4.6: Reuse Module
92	Figure4.7: Adaptation Module
93	Figure4.8: Evaluation Module
94	Figure4.9: Retain Module
96	Figure 4.10: Sample 1 of Case Design in Case Base
97	Figure 4.11 Sample 2 of Case Design in Case Base
97	Figure 4.12: Recall cases from case base
98	Figure 4.13: Recall case details
98	Figure 4.14: Case matching results for doctor references
99	Figure 4.15: Case matching results for patient references
101	Figure 5.1: Start Up Functions
102	Figure 5.2: Case matching Function
103	Figure 5.3: Matched Case Display Function for doctor and patient references
104	Figure 5.4: Cases Display Function
104	Figure 5.5: Case Details Display Function
105	Figure 5.6: New Cases Adding Function

106	Figure 5.7: Define Case Base
107	Figure 5.8: Define Attributes of Case
108	Figure 5.9: Define class and instance cases
109	Figure 5.10: Save Case Base
109	Figure 5.11: Restore Case Base
117	Figure 5.12: Feature weight of AHP model for Traditional Chinese Herbs
121	Figure 5.13: The decision tree of Traditional Chinese Herb
124	Figure 5.14: Overall similarity of Pure k-NN matching
124	Figure 5.15: Overall similarity of AHP weighted K-NN model matching
129	Figure 6.1: Samples of Art Script for Case Base reasoning.

LIST OF TABLES

PAGE	TABLE
10	Table 1.1: Gantt Chart of project schedule
40	Table 2.1: Technologies Comparisons (Ian Watson, 1997)
59	Table 2.2: Summary of CBR software tools
87	Table 4.1: Symbols in Data Flow Diagrams (DFD)
120	Table 5.1: Comparison between Nearest Neighbor Retrieval, AHP weighted k-NN retrieval and Inductive Retrieval
125	Table 5.2: Comparison Results of classification accuracy (Cheol-Soo Par*, Ingoo Han) between pure k-NN and AHP weighted k-NN

CHAPTER 1: INTRODUCTION

1.1 GENERAL VIEW

In this paper I present the result of a project on developing a knowledge-based system using Case-Based reasoning (CBR) for identifying purpose of traditional Chinese Herbs. The system is an interactive one which asks questions to an expert in traditional Chinese medical or traditional Chinese herbs and gradually narrows down the possibilities, eventually deciding the recommended Herbs as medicine.

CBR is one of the many technologies that are commonly known as expert or knowledge-based systems. It differs from the conventional way of how a computer solves a problem by solving problems using past experiences and it can support imprecise fuzzy queries. In addition, CBR facilitates the handling of voluminous quantities of data.

The goal of this project is to demonstrate the feasibility of the application of CBR technology in the areas of traditional Chinese medical and herbs.

1.2 DEFINITION

Artificial Intelligence (AI) simulates and applies human cognitive thinking toward problem solving. This project is build under the concept of developing a method for generate a combination of herbs for curing purpose with the aid of a Case-Based

Reasoning system. The knowledge or cases used in designing this system is formulated by traditional Chinese doctor. I have identified some of the benefits and uses of this Case-Based Reasoning system which includes using it in the medical field or using it for educational purposes.

One of the goals of AI is to simulate and apply a human's cognitive thinking ability towards solving problems. Often, researchers incorporate artificial intelligence within other disciplines (e.g. engineering, physical and life sciences, etc.) in order to develop automated systems for solving problems in those domains. With a huge array of traditional Chinese herbs being recognized in this world, it is extremely difficult for a person other than a traditional Chinese doctor or other expert to recommend a specific type of herbs as medicine according to the symptoms or illness given.

Sometimes, people especially Chinese prefer to take Chinese medicine for non heavy illness because Chinese medicine are believed to be more effective without bringing any side effects as western medicine does.

A doctor would have known the kind of illness the patient suffering through the doctor's knowledge and experiences but it would be very inconvenient for the doctor to memorize all the names of the herbs because in Chinese medical, quite a number of types of herbs will be needed to cure an illness. People will make mistake or forget no matter how experience they are. The doctor will have to spent most of his lifetime in memorizing the name of the herbs and this will make him neglected his researches in the medical field because it will be probably no time for researches to be done.

Keeping this problem in mind I wanted to develop an automated system which would encode a Traditional Chinese doctor's knowledge on how to identify some herbs and recommend it according to the illness, patient's bio-data and patient's conditions given, so that traditional Chinese doctor can use the system to make their job easier. This system will generate a list of suitable herbs for the patient by analyzing the illness the patient suffering from and the patient's physical and mental conditions

Exactly how to represent the various types of traditional herbs in a computer system? To begin with, the aforementioned expert formulates a set of keys that identify herbs based on their functions.

An example would be: The illness of Exterior can be categorized into Wind Chill or Wind Heat. Different combination of herbs will be used in different categories. Overdose of certain herbs like 'Magnoliae Liliflorae' may cause red eyes and dizziness for sensitive people. Certain Herbs are also toxic to some patient. So it is not suitable for sensitive patient and different herbs will be used to replace those herbs.

According to the research, I discovered that the process identifying herbs could be compiled in terms of cases. If compare to conventional rule-based system, Cases are independent from each other, maintenance of the CBR system can be done easily by just adding or deleting. Other than that, domain experts and novices can understand cases quite easy. For this reason I chose to represent the herbs-identification knowledge with the case-based knowledge representation scheme.

1.3 OBJECTIVE

There are some reasons or objectives which this system has been proposed:

a) **To create paperless working environment**

No more paper will be needed in recording the various types of herbs. The system will automatically generate the most suitable herbs for the patient.

b) **To minimum the human errors.**

Human often forget things and make mistakes. This system can avoid this kind of problems and more reliable.

c) **To provide easy to use and user friendly system**

This system will be used in medical field, which mean not all the users are familiar in using computer program. To encourage the used of computer, the user should be able to use the system within a short period of time.

d) **To make traditional Chinese medical a common**

Traditional Chinese medicine has already been recognized by most of the countries as an efficient treatment of several of illness. In Malaysia, the traditional Chinese medicine is still not very common among most of the non Chinese citizen. Most of them are still having the concept that the traditional Chinese medical are not as reliable as the western medical. By using this system,

I hope that it will make the traditional Chinese medical a common among all the Malaysian.

e) **To provide alternative way in getting medical advice**

Most of the people are not willing to spent money for non serious illness. They will probably buy the traditional herbs from the shops nearby according to the previous medical list given by the doctor. Does this really work? Different kind of herbs will be needed for different people and different condition for the same kind of illness. The doctor will know it through their knowledge and experiences. For the patient who not willing to spent money and time to see a doctor, this system will be an alternate way for them to get medical advices.

f) **Applying AI technique in medical field**

For many people, artificial intelligence or is still the stuff of science fiction. But for researchers specializing in case base reasoning systems technology, a sub-area of AI, it is very much a reality. This project is build practical knowledge based systems using Case-Based reasoning. Case-Based reasoning systems are computer systems that show intelligence or reasoning capabilities based on the knowledge supplied by domain experts so it works like an expert.

1.4 SCOPE

Given the wide scope of traditional Chinese medical and case-based reasoning from the research, the project will focus on a narrower perspective mainly in the domain of herbs.

- The system is designed for user who is sufficiently knowledgeable in the field of traditional Chinese medicine.
- The system will be able to provide treatment for illness that are already diagnosed by the traditional Chinese medical doctors
- Depending on the requirements of the treatment, it will query the user on the bio-data and current condition of the patient
- The system will store a brief record of patients like bio-data and prior consultation
- The system will be able to check for herbs interactions in the case where multiple herb are prescribed for multiple illness
- There will be an interaction among the herbs prescribed, it would be able to recommend an alternative herbs combination
- If a combination is not acceptable by the user, the system would be able to prescribe an alternative herbs combination, taking into account the herbs that are deemed acceptable and not acceptable by the user
- It will include an explanation facility as to how and why it was able to conclude with a particular herbs recommendation

- The system should contain some sample animation files and help file to help users to use that application system

1.5 PROJECT ASSUMPTION

The following are the assumptions made prior to beginning this project:

- The illness should have been already diagnosed by the traditional Chinese doctor
- Uses of the system have fundamental knowledge in the field of traditional Chinese medical and herbs, such as undergraduate in the field of traditional Chinese medicine, such as undergraduate traditional Chinese medical doctors

1.6 MOTIVATION FACTOR

- Shortage of recognize expert in traditional Chinese medical and many existing expert may not always available because of the constraints.
- Due to the unavailability of expert in the traditional Chinese medical field, immediate expertise was needed in possibly life threatening situations
- Time constrains required decisions to be made with limited or inexact information
- Existing solutions may be irrational in cases where traditional Chinese herbs recommendations were inappropriate for the problem

- Remembering the appropriate and possible of a large number and various type of herbs is a challenge for the traditional Chinese Doctors

Due to the above factors, there is a real demand for an intelligent system that would be able to capture the knowledge of current traditional Chinese medical and traditional Chinese herbs expert to be able to emulate them in the prescribing herbs for patients

1.7 SYSTEM LIMITATION

The following are the few unavoidable constrains that pose a limit on the system:
This system is not build for medical diagnosis. The system can not figure out what kind of illness does the patient suffering by analyzing the symptoms. It is just for the doctor to enter the name of illness and the system will generate a list of recommended traditional Chinese herbs as medicine.

The system is not a Chinese language system. That means the user have to translate the name of the illness and the patient's conditions into English language. Unfortunately most of the Chinese patients are used to Chinese language.

The system is not a full Chinese medical system. It does not provide any physical treatment for the patient. The system will just generate a list of recommended traditional Chinese herbs for the patient. For serious illness that need physical medical treatment like Acupuncture or needle treatment, the patient has to get the doctor advices.

This system can only generate herbs for some of the existing illness. A research has been done, and according to statistical report from the research, I can only analyze some of the common illness and its traditional Chinese treatment.

This system was developed with less consultation by doing reference from the Chinese medical book in the market. The knowledge gain from books is most probably limited and with unavailable bias.

1.8 SCHEDULE

Project scheduling is used to plan and control a project efficiently and can determine:

1. The minimal expected completion time for a project.
2. The delay of project activities.
3. The earliest and latest time each activity can started and completed.
4. The amount of slack for each activity.
5. Whether or not a current project is on schedule or is being completed within budget.

In this project, I have chosen to use Gantt chart to determine a clear timeline between starting date and the finishing date. An important advantage of this chart over other time-charting techniques is its simplicity. By using Gantt chart, a schedule of earliest possible project completion date.

There are six major phases in this project:

- Literature Review
- System Analysis
- System Design
- System Coding
- System Testing
- Documentation

Table 1.1: Gantt Chart of project schedule

Activity	Duration	Start	End	June	July	Aug	Sept	Oct
Literature Reviews	40Days	4 June 2002	14 July 2002					
System Analysis	40Days	15July 2002	24 Aug 2002					
System Design	30Days	25 Aug 2002	6 Oct 2002					
System Coding	90Days	7 Oct 2002	7 Feb 2002					
System Testing	40Days	15 Oct 2002	7 Feb 2002					
Documentation	180 Days	3 June 2002	7 Feb 2002					

1.9 REPORT LAYOUT

The following layout is to give an overview of the major phases involved during the development of the project. The purpose of this report is to document all the essential information gathered and used to develop this system. Below is the project report layout:

Chapter 1: Introduction

This chapter serves as an introduction to the entire project. It made an overview on the, definition, project objectives, project scopes, project assumptions, motivation factors, project limitation and the project schedule.

Chapter 2: Literature Review

This chapter covers all the literatures survey done on this project, including reviews on the features, capabilities, system architecture, system designing tool and so on.

Chapter 3: Methodology

This chapter fairly discusses the development methodology, the functional and non-functional requirements, and also the tools and technology consideration of this project.

Chapter 4: System Design

This chapter describes the design considerations including processing design, the user interface design and also the case base design of this project.

1.10 SUMMARY

This chapter focuses mainly on the introduction of this project. The first part is the general view and definition about this project. Then the objectives for Case-Based Reasoning system for the usage of identifying traditional Chinese Herbs were discussed and determined; and the scope of this project is also defined in section 1.4.

Secondly, section 1.5 shows the project assumption and description of motivation factor is showed in section 1.6. Section 1.7 shows system limitations of those functions which most probably unable to achieve in this project. Finally, there is a project schedule shown by the Gantt chart in section 1.8 to tell the project's timeline and development; and the project report layout was shown in the section 1.9.

CHAPTER 2: LITERATURE

Research and literature reviews were the tasks to be done in this chapter. Effort to dig deeper for information about the proposed system is important. With that, the overview of the system will be seen. The materials below are concluded following some efforts such as studying and looking for the similar projects from FSKTM document room, looking for reference books from library, extra reference materials from Internet and purchasing some useful related books.

There are three parts of literature review approaches, findings and analysis. Firstly, literature review will identify all the approaches that were used to find information for the project and then provide sources and summary of the findings. Finally to analyze the strengths and weakness of the information that was founded.

2.1 INFORMATION SOURCE AND FINDING

Many approaches to find information on Case-Based Reasoning (CBR) and Traditional Chinese Herbs had been utilized. The available information on CBR is not that much but there is much information about Traditional Chinese Herbs as many researches have been done in others countries. My finding sources are generally divided into few categories:

- Internet
- Article for media
- Group discussion
- Others

2.1.1 Internet

Internet will play the major role in finding all the information about Case-Based Reasoning Traditional Chinese Herbs, Knowledge-Based system and others. Internet can provide me up-to-date information and this is important because we are in the technology world that everything has kept updating. There are plenty of CBR researches available in the Internet and CBR applications are very wide indeed. As a research, I have downloaded many articles about CBR developing system in order to better understanding of CBR.

2.1.2 Articles from Medias

- Newspaper
- In-Tech from star
- Compute Time from News Strait Times

- PC World

2.1.3 Group Discussions

There are few of them who work on Case-Based Reasoning system which I can cooperate with and we learn form each other to make our project better. Beside that our discussions save quite of our time to do the unnecessary things.

2.1.4 Others

My Supervisor Assoc. Prof. Dr. Syed Malek Fakar Duani has kindly shared his research on CBR system with us to make our research easier and faster. Other than that, studies have been done through similar works made of seniors in the FSKTM documents room. Although there are no exactly similar work has been found, but I study about how they express CBR for a better understand on my CBR application. Beside that, I had found more CBR application reference website from their work.

2.2 ARTIFICIAL INTELLIGENCE

Artificial Intelligence (AI) is a combination of computer science, physiology, and philosophy. AI is a broad topic, consisting of different fields, from machine vision to expert systems. The element that the fields of AI have in common is the creation of machines that can "think".

Research into the areas of learning, language, and of sensory perception has aided scientists in building intelligent machines. One of the most challenging approaches facing experts is building systems that mimic the behavior of the human brain, made up of billions of neurons, and arguably the most complex matter in the universe. Perhaps the best way to gauge the intelligence of a machine is British computer scientist Alan Turing's test. He stated that a computer would deserve to be called intelligent if it could deceive a human into believing that it was human.

Artificial Intelligence has come a long way from its early roots, driven by dedicated researchers. The beginnings of AI can reach back before electronics to philosophers and mathematicians such as Boole and others theorizing on principles that were used as the foundation of AI Logic. AI really began to intrigue researchers with the invention of the computer in 1943. The technology was finally available, or so it seemed, to simulate intelligent behavior. Over the next four decades, despite many stumbling blocks, AI has grown from a dozen researchers, to thousands of engineers and specialists; and from programs capable of playing checkers, to systems designed to diagnose disease.

AI has always been on the pioneering end of computer science. Advanced-level computer languages, as well as computer interfaces and word-processors owe their existence to the research into artificial intelligence. The theory and insights brought about by AI research will set the trend in the future of computing. The products available today are only bits and pieces of what are soon to follow, but they are a movement towards the future of artificial intelligence.

2.2.1 Introduction to Artificial Intelligence

AI can be broken into several different disciplines. They are each unique, but often they intermix to accomplish a programming task, and the differences become fuzzy. The different disciplines are expert systems, natural languages, simulation of human sensory capabilities, robotics, and neural networks.

Expert Systems: Expert systems are also known as knowledge-based systems. They are computer systems that rely on a knowledge base of rules that pertain to a specific application area. Experts in the application area give rules of thumb to use in certain situations. The rules of thumb are linked into preset if-then rules to solve the problem. Essentially, the expert system mimics the expert's thought process in troubleshooting the problem.

Expert Systems are able to solve problems. Solving problems is their intention. They are able to break a large problem into smaller parts in order to come to a solution. They understand the information given to them by the user. If the information is contradictory

or ambiguous, the expert system asks for clarification or more information. Expert systems can learn. If an expert system is faced with a new problem, it will save the information and the solution the user chooses for future use.

Natural Language Systems: Natural language systems are systems that enable computers to accept, interpret, and execute instructions in the natural language of the user. The intent of natural language systems is to create a more natural interaction between human users and computers. Database query is the area that has benefited the most from natural language systems. Other areas appropriate for natural language include machine translation, summarizing and searching bibliographic texts, and analyzing style and sentence structure.

The biggest contribution to the evolution of machine intelligence by natural language and conversation systems is in the interaction between humans and machines.

Human Sensory Simulation: Simulation of human sensory capabilities focuses on seeing, hearing, speaking, and touching. I will focus on two systems: voice recognition and vision input systems.

- **Voice recognition systems** - allow the user to speak to the computer. Vision input systems are used to enable computers to see and understand its environment.
- **Vision input systems** - are still in their infancy. The storage space and processing speed needed for a fully functional vision input system are too restrictive. The human vision process is extremely complex. One commercial use

of visual input systems is in assembly line inspection robots. This type of system will have a camera placed over the assembly line to inspect parts for defects. The digitized camera image is compared to an image in a database. If a defect is found, the system alerts the main assembly computer to bring it to the human operator's attention, and appropriate action can be taken.

Robotics: Robotics is the integration of computers and robots. Robots are taught to perform repetitive tasks. Intelligent robots incorporate the other disciplines of AI. They use human sensory simulation for touch, sight, and hearing. A useful robot in use is a security robot for warehouses. They are given a internal map of the area they are to patrol. They listen for abnormal sounds, and look for intruders or fire. If an abnormality is found, they contact the authorities and employees. Another useful robot collects aluminum cans and trash in office buildings after everyone has left for the day.

The basis of a robot's intelligence is in its programming. A knowledge base may be used for the robot to figure out what to do if it finds a fire. They must be aware of their surroundings, and where they are in their surroundings. They will faithfully serve their particular purpose.

Neural Networks: Neural networks are a fairly new addition to the field of artificial intelligence. Neural networks are a simulation of the processes of the human brain. As Rao (1995) explain, "The human brain uses a web of interconnected processing elements called neurons to process information."

Simply, neural networks are a system of interconnected computing elements. The individual elements are data objects commonly referred to as nodes or neurons. A weighted signal is sent into the network. Each node has only one output based on its input. The input spreads through the network forming a pattern. Information is stored as the pattern of interconnection that is formed between the nodes.

The behavior of a neural network system is more human than conventional computer systems. Rosenfeld states, "They learn, forget, and most importantly, they can be structured so that they self-organize to represent information and solve problems." (1995). Neural networks are not very good at solving simple problems. They are best at solving complex problems that can not be solved by a simple algorithm. They are also very good at solving problems whose input is noisy.

Neural networks can be built to solve specific problems, but they are most useful for their ability to learn. A string of inputs is introduced to the network along with the desired result. The network will form the solution. For this reason neural networks are the masters of pattern recognition.

Incorporate fuzzy logic and this ability is increased ten fold.

The major fields of interest for neural networks make use of the neural networks ability to filter noise and recognize patterns. These fields are handwriting and speech recognition, and predicting stock patterns. Neural networks are also found in fields such as finance, reading IRS tax forms, defense systems, and vehicle control.

2.2.2 The History of Artificial Intelligence

Evidence of Artificial Intelligence folklore can be traced back to ancient Egypt, but with the development of the electronic computer in 1941, the technology finally became available to create machine intelligence. The term artificial intelligence was first coined in 1956, at the Dartmouth conference, and since then Artificial Intelligence has expanded because of the theories and principles developed by its dedicated researchers. Through its short modern history, advancement in the fields of AI have been slower than first estimated, progress continues to be made. From its birth 4 decades ago, there have been a variety of AI programs, and they have impacted other technological advancements.

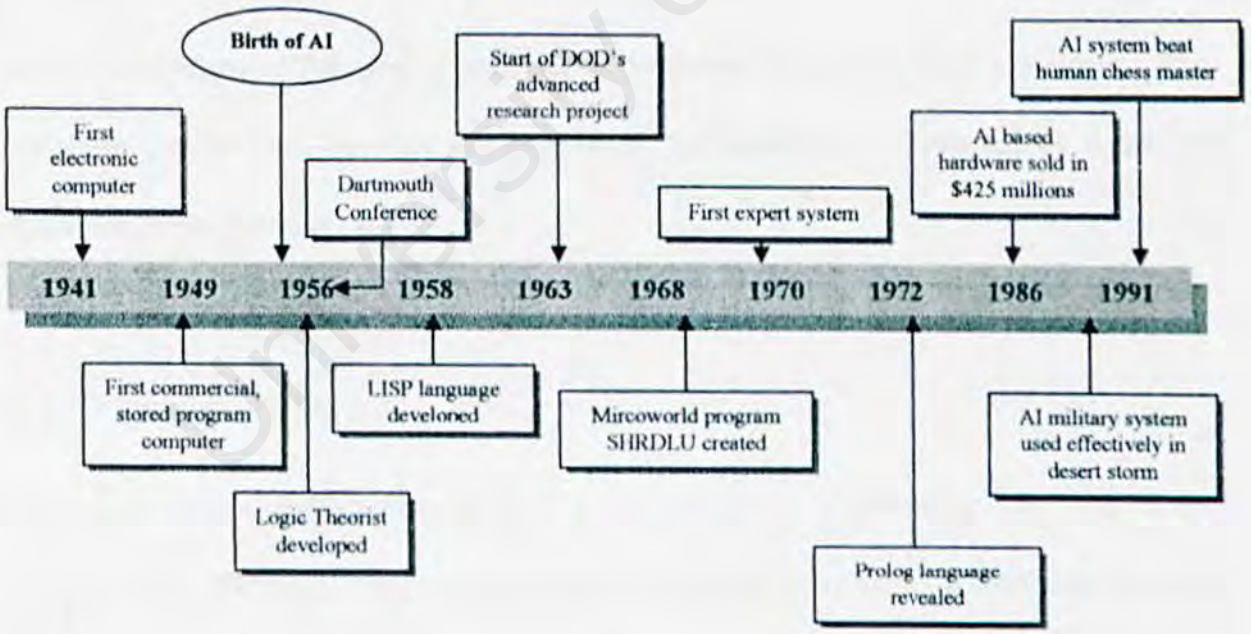


Figure 2.1: Timeline of major AI event

2.3 CASE-BASED REASONING

What is Case-Based Reasoning (CBR)? There are many definitions for CBR:

- Case-Based Reasoning is reasoning by remembering (Leake, 1996)
- A case-based reasoner solve new problems by adapting solutions that were to solve old problems (Riesbeck and Schank, 1989)
- Case-based reasoning is a recent approach to problem solving and learning (Aamodt & Plaza, 1994)
- Case-based reasoning is both the ways people use cases to solve problems and the ways we can make machines use them (Kolodner,1993)

More specifically, CBR uses a database of problems to resolve new problems. The database can be built through the knowledge engineering (KE) process or it can be collected from previous cases.

2.3.1 History of CBR

The work Schank and Abelson in 1977 is widely held to be the origins of CBR. They proposed that our general knowledge about situations is recorded as scripts that allow us to set up expectations and perform inferences. Scripts were proposed as a structure for conceptual memory describing information about stereotypical events such as, going to a restaurant or visiting a doctor. However, experiments on scripts showed that they were

not a complete theory of memory representation because people often confused events that had similar scripts. For example, a person might mix up room scenes from a visit to a doctor's office with a visit to a dentist's office. Such observations fell in line with the theories of concept formation, problem solving and experiential learning within philosophy and psychology.

Roger Schank continued to explore the role that the memory of previous situations (i.e., cases) and situation patterns or *memory organization packets* (MOPs) play in both problems solving and learning. At a similar time Gentler was developing a theoretical framework for analogy which also has relevance to CBR. Perhaps with the benefit of hindsight it is also possible to find references of significance to CBR in Wittgenstein's observation that natural concepts such as tables and chairs are in fact polymorphic and can not be classified by a single set of necessary and sufficient features but instead can be defined by a set of instances (i.e., cases) with family resemblances. This work has been cited by Aamodt and Plaza [94] as a philosophical basis for CBR.

Whilst the philosophical roots of CBR could perhaps be claimed by many what is not in doubt is that it was the work of Roger Schank's group at Yale University in the early eighties that produced both a cognitive model for CBR and the first CBR applications based upon this model. Janet Kolodner developed the first CBR system called CYRUS. CYRUS contained knowledge, as cases, of the travels and meetings of ex-US Secretary-of-State Cyrus Vance. CYRUS was an implementation of Schank's dynamic memory model. Its case-memory model later served as the basis for several other CBR systems including MEDIATOR, CHEF, PERSUADER, CASEY and JULIA.

An alternative approach came from Bruce Porter's work, at The University of Texas in Austin, into heuristic classification and machine learning resulting in the PROTOS system. PROTOS unified general domain knowledge and specific case knowledge into a single case memory model. This work was taken further by GREBE, a system operating in a legal domain.

It is perhaps no surprise that since the practice of law is largely based upon precedence and the notion of cases, that there has been some interest from this sector in CBR. Edwina Rissland is a group at the University of Massachusetts in Amherst who developed HYPO. In HYPO cases representing legal precedents are used to interpret a court situation and to produce arguments for both the defense and the prosecutors. This system was later combined with rule-based reasoning to produce CABARET.

CBR research is not restricted to the US, but it was slower to get started in Europe. Amongst the first cited European work is that of Derek Sleeman's group from Aberdeen in Scotland. They studied the uses of cases for knowledge acquisition, developing the REFINER system. At a similar time Mike Keane, from Trinity College Dublin, undertook cognitive science research into analogical reasoning that has subsequently influenced CBR.

On continental Europe Michael Richter and Klaus Althoff from the University of Kaiserslautern applied CBR to complex diagnosis. This has given rise to the PATDEX system and subsequently to the CBR tool S3-Case. Agnar Aamodt at the University of Trondheim has investigated the learning facet of CBR and the combination of cases and general domain knowledge resulting in CREEK.

In the UK, CBR seems to be particularly applied to civil engineering. A group at the University of Salford is applying CBR techniques to fault diagnosis, repair and refurbishment of buildings. Yang & Robertson in Edinburgh are developing a CBR system for interpreting building regulations, a domain reliant upon the concept of precedence. Whilst another group in Wales applying CBR to the design of motorway bridges.

Further a field there is active CBR groups in Israel, India and Japan. However, the increasing number of CBR papers in AI journals and the increasing number of commercially successful CBR applications is likely to ensure that many more countries take an active interest in CBR in the future. As an indicator the British Computer Society Specialist Group on Expert Systems has held CBR workshops suitable for novices at both it last annual conferences.

2.3.2 Description of CBR

CBR is exactly how the real world people solve problems by assuming the following conditions:

- Regularity - the world is essentially a regular and predictable place. The same actions performed under the same conditions will normally have the same or very similar outcomes.
- Typicality - events tend to repeat. Thus a CBR system's experiences are likely to be useful in the future.

- Consistency - small changes in the world only require small changes to our reasoning and need correspondingly small changes to our solutions.

These points make CBR a useful approach for designing a system for a problem that is regular, consistent and predictable, in which things learned in the past will be useful in similar circumstances in the future.

CBR will make use of the experience captured in this case base to solve new diagnostic problems. When encounter a new, unsolved problem, a past case that is similar to that new problem will very likely contain an appropriate maintenance operation.

Consider a simple example of CBR that deals with car diagnostics. A case stored in the case base is a fault that has been solved in the past. The case description is made up of effects, such as observed symptoms (e.g., engine does not start) and context parameters (e.g., ignition key is turned on). It can also include measured parameters for example, the state of the electronic control units obtained using testing equipment. The solution is the maintenance operation.

2.3.3 How CBR Work?

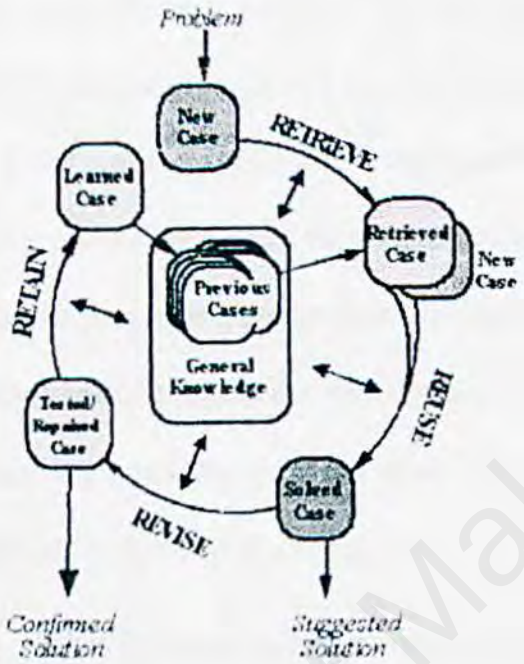


Figure2.2: CBR Cycle (Aamodt & plaza, 1994, AI Communication)

The CBR process can be represented by a schematic cycle, as shown in Figure 2.1. Conceptually, the CBR solves problems following the four phases (Aamodt and Plaza, 1994):

- Retrieve the most similar cases(s)
- Reuse the case(s) to attempt to solve the problem
- Revise the proposed solution if necessary.
- Retain the new solution as a part of a new case.

During retrieval the most similar case or cases in the case base are determined, based on the new problem description. During reuse the information and knowledge in the retrieved case(s) is used to solve the new problem. The new problem description is combined with the information contained in the old case to form a solved case. During revision the applicability of the proposed solution is evaluated in the real world. If necessary and possible, the proposed case must be adapted in some way. If the case solution generated during the revise phase must be kept for future problem solving, the case base is updated with a new learned case in the retain phase.

CBR is concerned with assessing similarity between cases. Therefore, the higher the number of cases, the more powerful is the CBR system.

The RETRIEVE process in CBR is different from the process in a database. When user want to query data, the database only retrieves some data using an exact matching while a CBR can retrieve data using an approximate matching.

As shown in Figure 2.1, the CBR cycle starts with the description of a new problem, which can be solved by retrieving previous cases and reusing solved cases, if possible, giving a suggested solution or revising the solution, retaining the repaired case and incorporating it into the case base. However, this cycle rarely occurs without human intervention that is usually involved in the RETAIN step. Many application systems and tools act as a case retrieval system, such as some help desk systems and customer support systems.

CBR systems store knowledge in four different knowledge containers (Richter, 1995):

- Vocabulary

- Case base
- Similarity assessment
- Solution adaptation

The following describe the CBR system in more details:

Case Representation

A case is a contextualized piece of knowledge representing an experience. It contains the past lesson that is the content of the case and the context in which the lesson can be used. Typically a case comprises:

- Problem that describes the state of the world when the case occurred
- Solution which states the derived solution to that problem
- Outcome which describes the state of the world after the case occurred

Cases which comprise problems and their solutions can be used to derive solutions to new problems. Cases comprising problems and outcomes can be used to evaluate new situations. If, in addition, such cases contain solutions they can be used to evaluate the outcome of proposed solutions and prevent potential. Cases can be represented in a variety of forms using the full range of AI representational formalisms including frames, objects, predicates, semantic nets and rules - the frame/object representation currently being used by the majority of CBR software.

There is a lack of consensus within the CBR community as to exactly what information should be in a case. However, two pragmatic measures can be taken into account in deciding what should be represented in cases: the functionality and the ease of acquisition of the information represented in the case.

Case Indexing

Case indexing involves assigning indices to cases to facilitate their retrieval. Several guidelines on indexing have been proposed by CBR researchers. Indices should be:

- Predictive,
- Address the purposes the case will be used
- Abstract enough to allow for widening the future use of the case-base
- Concrete enough to be recognized in future

Both manual and automated methods have been used to select indices. Choosing indices manually involves deciding a case's purpose with respect to the aims of the reasoner and deciding under what circumstances the case will be useful.

There are an ever increasing number of automated indexing methods including:

- Indexing cases by features and dimensions that tend to be predictive across the entire domain i.e., descriptors of the case which are responsible for solving it or which influence its outcome. In this method the domain is analyzed and the

dimensions that tend to be important are computed. These are put in a checklist and all cases are indexed by their values along these dimensions.

- Difference-based indexing selects indices that differentiate a case from other cases. During this process the system discovers which features of a case differentiate it from other similar cases, choosing as indices those features that differentiate cases best.
- Similarity and explanation-based generalization methods, which produce an appropriate set of indices for abstract cases created from cases that share some common set of features, whilst the unshared features are used as indices to the original cases.
- Inductive learning methods, which identify predictive features that are then used as indices.
- Explanation-based techniques, which determine relevant features for each case. This method analyses each case to find which of their features predictive ones are. Cases are then indexed by those features.
- However, despite the success of many automated methods, Janet Kolodner believes that people tend to do better at choosing indices than algorithms, and therefore for practical applications indices should be chosen by hand.

CBR Storage

Case storage is an important aspect in designing efficient CBR systems in that, it should reflect the conceptual view of what is represented in the case and take into account the

indices that characterize the case. The case-base should be organized into a manageable structure that supports efficient search and retrieval methods. A balance has to be found between storing methods that preserve the semantic richness of cases and their indices and methods that simplify the access and retrieval of relevant cases. These methods are usually referred to as **case memory models**. The two most influential case memory models are the **dynamic memory model** and the **category-exemplar model**.

- ***The dynamic memory model***

The case memory model in this method is comprised of memory organization packets (MOPs). MOPs are a form of frame and are the basic unit in dynamic memory. They can be used to represent knowledge about classes of events using two kinds of MOPs:

- *Instances* representing cases, events or objects
- *Abstractions* representing generalized versions of instances or of other abstractions.

The case memory, in a dynamic memory model, is a hierarchical structure of episodic memory organization packets (E-MOPs), also referred to as generalised episode (GEs) developed from Schank's more general MOP theory. The basic idea is to organize specific cases which share similar properties under a more general structure. A GE contains three different types of objects: *norms*, *cases* and *indices*. Norms are features common to all cases indexed under a GE. Indices are features which discriminate between a GE's cases. An index may point to a

more specific generalized episode or to a case, and is composed of an index name and an index value.

The case-memory is a discrimination network where nodes are either a GE, an index name, index value or a case. Index name-value pairs point from a GE to another GE or case. The primary role of a GE is as an indexing structure for storing, matching and retrieval of cases. During case storage when a feature (i.e., index name and index value) of a new case matches a feature of an existing case a new GE is created. The two cases are then discriminated by indexing them under different indices below the new GE (assuming the cases are not identical). Thus, the memory is dynamic in that similar parts of two cases are dynamically generalized into a new GE, the cases being indexed under the GE by their differences.

However, this process can lead to an explosive growth in the number of indices as case numbers increase. So for practical purposes most CBR systems using this method limit the number of permissible indices to a limited vocabulary.

- ***The category-exemplar model***

This model organizes cases based on the view that the real world should be defined extensionally with cases being referred to as exemplars. The case memory is a network structure of categories, semantic relations, cases and index pointers. Each case is associated with a category. Different case features are assigned different

importance in describing a case's membership to a category. Three types of indices are provided, which may point to a case or a category:

- *Feature links* - point from problem descriptors (features) to a case or category,
- *Case links* - point from categories to its associated cases, and
- *Difference links* - pointing from categories to the neighboring cases that only differ in a small number of features.

A feature is described by a name-value pair. A category's exemplars are stored according to their degree of prototypically to the category. Within this memory organization, the categories are inter-linked within a semantic network containing the features and intermediate states referred to by other terms. This network represents a background of general domain knowledge that enables explanatory support to some CBR tasks. A new case is stored by searching for a matching case and by establishing the relevant feature indices. If a case is found with only minor differences to the new case, the new case may not be retained, or the two cases may be merged.

Retrieval

Given a description of a problem, a retrieval algorithm, using the indices in the case-memory, should retrieve the most similar cases to the current problem or situation. The

retrieval algorithm relies on the indices and the organisation of the memory to direct the search to potentially useful cases.

The issue of choosing the *best* matching case has been addressed by research into analogy. This approach involves using heuristics to constrain and direct the search. Several algorithms have been implemented to retrieve appropriate cases, for example: serial search, hierarchical search and simulated parallel search.

Case-based reasoning will be ready for large scale problems only when retrieval algorithms are efficient at handling thousands of cases. Unlike database searches that target a specific value in a record, retrieval of cases from the case-base must be equipped with heuristics that perform partial matches, since in general there is no existing case that exactly matches the new case. The following are among well known methods for case retrieval:

- Nearest neighbor
- Induction
- Knowledge guided induction
- Template retrieval

These methods can be used alone or combined into hybrid retrieval strategies.

i) Nearest neighbor

This approach involves the assessment of similarity between stored cases and the new input case, based on matching a weighted sum of features. The biggest problem here is to determine the weights of the features. The limitation of this approach includes problems in converging on the correct solution and retrieval times. In general the use of this method leads to the retrieval time increasing linearly with the number of cases. Therefore this approach is more effective when the case base is relatively small.

ii) Induction

Induction algorithms determine which features do the best job in discriminating cases, and generate a decision tree type structure to organize the cases in memory. This approach is useful when a single case feature is required as a solution, and where that case feature is dependent upon others.

iii) Knowledge guided induction

This method applies knowledge to the induction process by manually identifying case features that are known or thought to affect the primary case feature. This approach is frequently used in conjunction with other techniques, because the explanatory knowledge is not always readily available for large case bases.

iv) Template retrieval

Similar to SQL-like queries, template retrieval returns all cases that fit within certain parameters. This technique is often used before other techniques, such as nearest neighbor, to limit the search space to a relevant section of the case-base.

Adaptation

Once a matching case is retrieved a CBR system should adapt the solution stored in the retrieved case to the needs of the current case. Adaptation looks for prominent differences between the retrieved case and the current case and then applies formulae or rules that take those differences into account when suggesting a solution. In general, there are two kinds of adaptation in CBR:

- **Structural adaptation** - Adaptation rules are applied directly to the solution stored in cases.
- **Derivational adaptation** - Reuses the algorithms, methods or rules that generated the original solution to produce a new solution to the current problem. In this method the planning sequence that constructed that original solution must be stored in memory along with the solution. Derivational adaptation, sometimes referred to a reinstantiation, can only be used for cases that are well understood.

An ideal set of adaptation rules must be strong enough to generate complete solutions from scratch, and an efficient CBR system may need both structural adaptation rules to adapt poorly understood solutions and derivational mechanisms to adapt solutions of

cases that are well understood. Several techniques, ranging from simple to complex, have been used in CBR for adaptation. These include:

- ***Null adaptation*** - A direct simple technique that applies whatever solution is retrieved to the current problem without adapting it. Null adaptation is useful for problems involving complex reasoning but with a simple solution.
- ***Parameter adjustment*** - *A* structural adaptation technique that compares specified parameters of the retrieved and current case to modify the solution in an appropriate direction.
- ***Abstraction and respecialisation*** - A general structural adaptation technique that is used in a basic way to achieve simple adaptations and in a complex way to generate novel, creative solutions.
- ***Critic-based adaptation*** - In which a critic looks for combinations of features that can cause a problem in a solution. Importantly, the critic is aware of repairs for these problems.
- ***Reinstantiation*** – It is used to instantiate features of an old solution with new features.
- ***Derivational replay*** - It is the process of using the method of deriving an old solution or solution piece to derive a solution in the new situation.
- ***Model-guided repair*** - Uses a causal model to guide adaptation, which is used for diagnosis and learning in auto mechanics, and used in the design of physical devices.
- ***Case-based substitution*** - Uses cases to suggest solution adaptation.

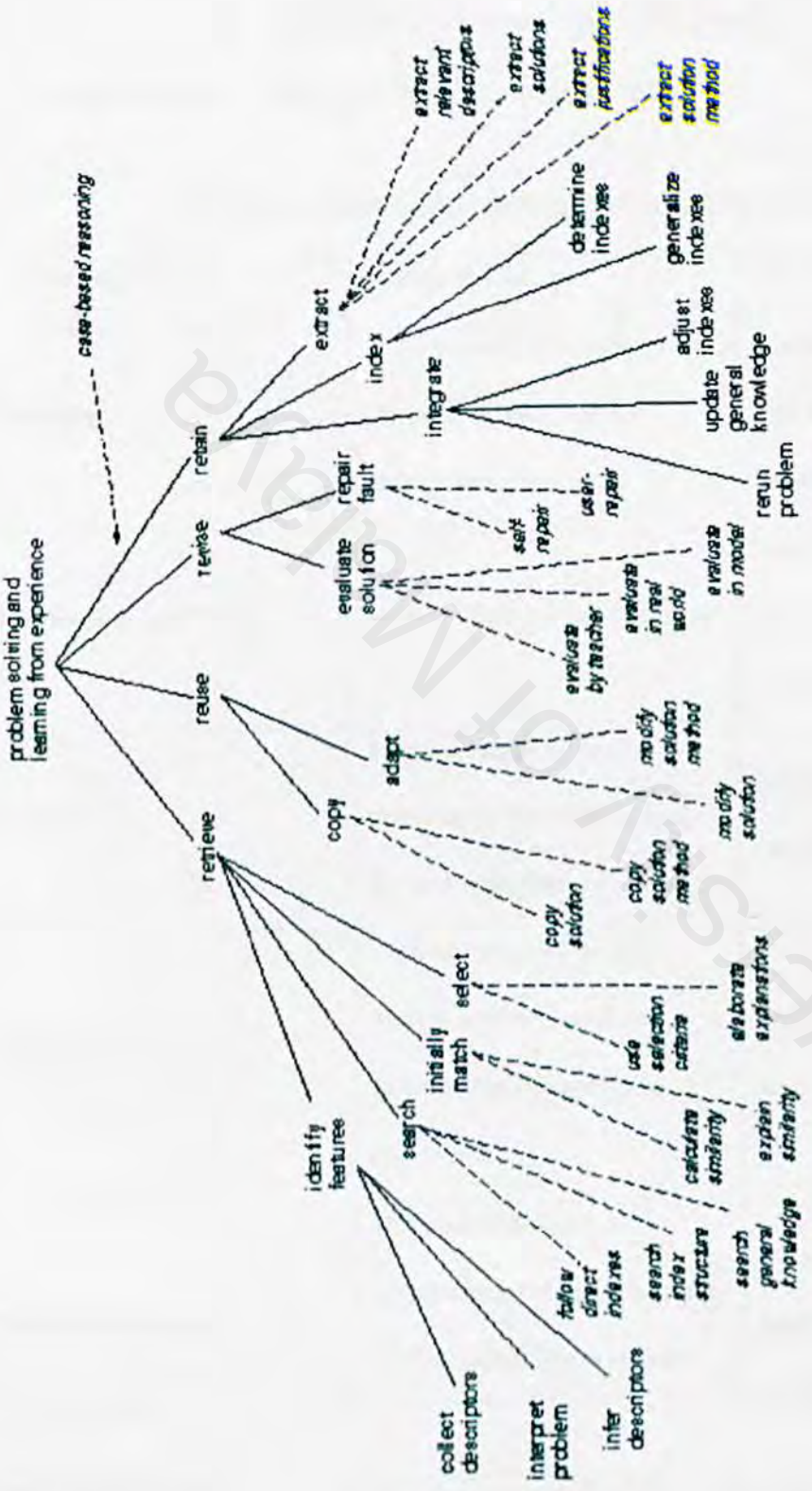


Figure 2.3: A Task-method Decomposition of CBR

2.3.4 CBR and Other Techniques

Every type of technology has got advantages and disadvantages. The table below shows the appropriate time in using different forms of systems.

Table 2.1: Technologies Comparisons (Ian Watson, 1997)

Technology Type	When to use	When not to use
Databases	Well-structured, standardized data and simple precise queries possible	Complex, poorly structured data and fuzzy queries required
Information retrieval	Large volumes of textual data	Non-textural complex data types, background knowledge available
Statistics	Large volumes of well-understood data with a well-formed hypothesis	Exploratory analysis of data with dependent variable
Rule-based systems	Well-understood, stable, narrow problem area and justification by rule-trace acceptable	Poorly understood problem area that constantly changes
Machine learning	Generalisable rules are required from a large training set and justification by rule-trace is acceptable	Rules are not required, and justification by rule-trace is unacceptable
Neural Networks	Noisy numerical data for	Complex symbolic data or

	pattern recognition or signal processing	when a justification is required
Case-based reasoning	Poorly understood problem area with complex structured data that changes slowly with time and justification required	When case data is not available, or complex adaptation is required, or if an exact optimum answer is required

2.3.5 CBR Applications

The applications of CBR are divided into two sub-sections. The first CBR applications are primarily the product of academic research. These systems, mostly developed in the US, demonstrate certain features of CBR. They were viewed as demonstrators, as there is little evidence they have been used in real situations.

Although CBR is a relatively new method there have been several successful commercial applications. The second sub-section discusses three such applications. The first developed at Lockheed is usually cited as the first commercial CBR application. Lockheed's CLAVIER system has already become the classic CBR system demonstrating that CBR could be applied to solve a problem where no explicit model existed, and can learn by acquiring new cases and so improve their performance with time.

The second developed by British Airways shows how CBR techniques are easily accepted by users because of CBR's intuitive feel and how retaining the rich context of a case has advantages over the distillated knowledge associated with rule-based systems. The final system developed for Legal & General shows how CBR systems can be implemented rapidly and can be maintained by users.

2.3.5.1 Classification of CBR Applications

CBR applications can be classified into two categories:

Analytic tasks – Focusing on analyzing situation where classification on the situation always involved. A new case is matched against those in the case-base from which an answer can be given. The solution from the best matching case is then reused. In fact, most commercial CBR tools support analytic tasks.

Synthesis tasks - Attempt to get a new solution by combining previous solutions and there are a variety of constraints during synthesis. Usually, they are harder to implement. CBR systems that perform synthesis tasks must make use of adaptation and are usually hybrid systems combining CBR with other techniques [Watson, 1997]. Synthesis task usually will infinite (or at least very large) solution space.

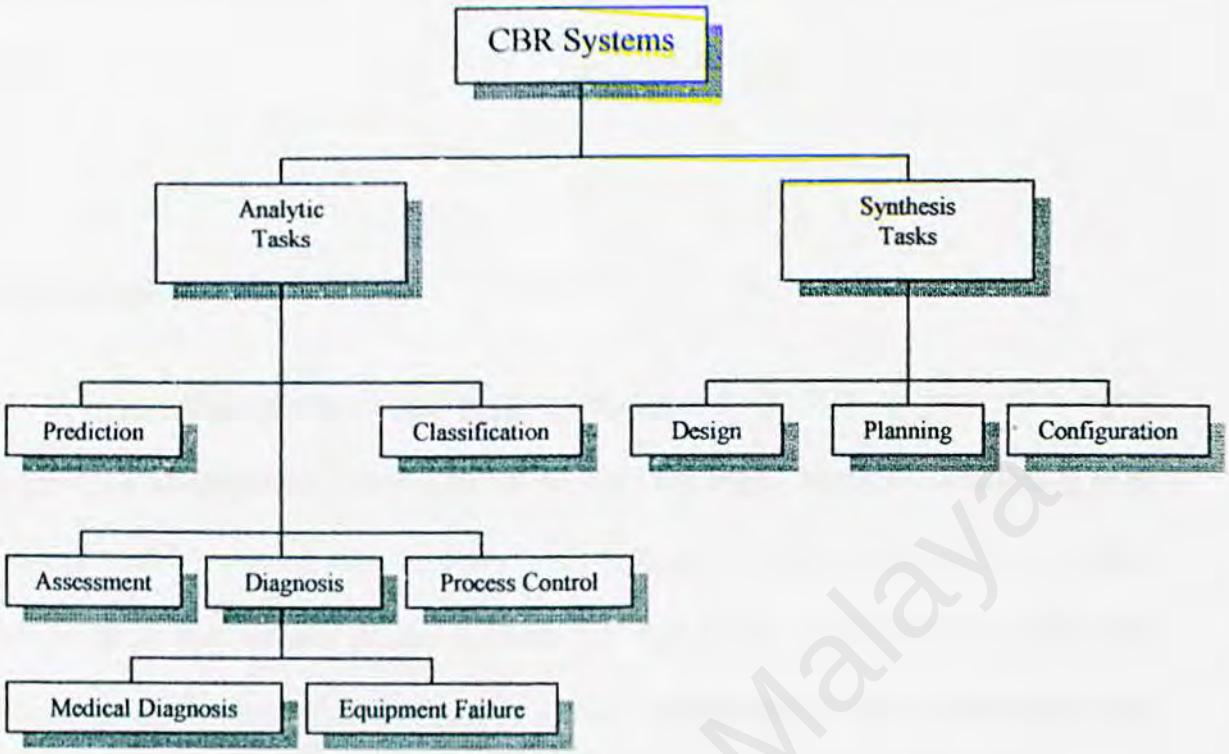


Figure 2.4: A Classification Hierarchy of CBR Applications [Althoff et al., 1995]

2.3.5.2 Implementation of Academic Research

The following is an indication of the range of problems to which CBR has been applied:

Knowledge acquisition

The REFINER program is a knowledge acquisition system that helps an expert refine his knowledge in a more natural way than extracting rules from an expert. The system has the ability to recognize classifications that are inconsistent and to suggest ways of

resolving the inconsistency. REFINER uses both inferred rules as well as individual cases.

Legal reasoning

JUDGE represents a case-based model of criminal sentencing. The program starts with a simple set of strategies for forming sentences and then begins to retrieve reminding of its own cases for developing new sentences. JUDGE can be used to reason about murder, man-slaughter and assault cases. JUDGE has five stages of operations which are: interpretation, retrieval, differential analysis, application and modification and generalization. JUDGE uses its case-base to maintain a consistent sentencing pattern.

Explanation

SWALE is a case-based creative explainer. It has a library of cases for explaining why animals and people die. If SWALE is given an anomalous event such as the death of a healthy, young horse, it searches for explanations of death in other context such as rock stars dying from drug overdoses or a spouse being murdered for life insurance, and then SWALE tries to adapt those explanations to fit the current situation.

Diagnosis

CASEY is a system which diagnosis heart failure. As input it uses a patient's symptoms and produces a causal network of possible internal states that could lead to those symptoms. When a new case arises CASEY tries to find cases of patients with similar but not necessarily identical symptoms. If the new case matches, then CASEY adapts the retrieved diagnosis by considering differences in symptoms between the old and new cases.

Arbitration

MEDIATOR works in the domain of dispute resolution. Given a conflict between several parties, it proposes possible compromises. If one proposal fails to satisfy all the parties involved, it generates new proposals and records the failure thus avoiding a similar failure in the future.

Design

JULIA is a case-based designer that works in the domain of meal planning. It uses cases to propose solutions, decomposing the problem as necessary and posting constraints to guide synthesis. It exploits a repertoire of adaptation methods to transform previous dishes in order to meet the constraints of the current problem. These adaptation methods are used both to modify previous cases and repair previous decisions that have been invalidated by constraints that arrive late.

CADET is a case-based design system that functions as designer's assistant for mechanical design. It retrieves previous successful designs while avoiding previous failures such as poor materials or high costs. CADET transforms abstract descriptions of the desired behavior of the device into a description that can be used to retrieve relevant designs and generate a variety of equivalent alternative designs for a given set of design specifications.

Planning

BATTLE projects the results for plan evaluation in the domain of land warfare planning. The system was built from an existing database of 600 cases. The user describe a battle situation and chosen battle plan, BATTLE retrieve cases that are composed of pieces of battles and experts' evaluations, to project the outcome.

Repair and adaptation

CHEF creates new recipes from old ones. CHEF begins planning by finding a recipe that satisfies as many of its active goals as possible. It uses a set of object critics and modification rules to change the old recipe and satisfy the goal of the new one. One of the important aspects of CHEF is explanation of failures through a causal description of why they occurred. CHEF links recipes to a failure's explanation to predict the failure in similar circumstances. CHEF stores new recipes indexed by the goals that they satisfy and the problems that they avoid.

2.3.6 Advantages of CBR

CBR provides a number of advantages over alternative approaches:

- Does not require extensive analysis of domain knowledge. CBR permits problem solving even if domain knowledge is incomplete. The most important thing is to know how to compare two cases.
- Allows shortcuts in reasoning. If a suitable case is found, a solution can be proposed quickly.
- Lead to improved explanation capability in situations where the most comprehensible explanations are those that involve specific instances [Branting and Aha, 1995].
- Help avoid past errors and learn from the errors and success. In CBR, the system keeps a record of each situation that occurred for future reference.
- High user acceptance.
- Make use of existing data, e.g. Database.
- Reduce the knowledge acquisition effort. In traditional knowledge-based system, acquisition of general knowledge is difficult. CBR system requires less general knowledge because most general knowledge is in case base. Case knowledge is easier to acquire.
- Required less maintenance effort. This is because cases are independent from each other. Furthermore Domain expert and novices understand cases quite easy. Maintenance of CBR system can be done by just adding deleting cases.

2.3.7 Disadvantages of CBR

- A case-based reasoner might be tempted to use old cases blindly, relying on previous experience without validating it in the new situation.
- A case-based reasoner might allow cases to bias him or her or it too much in solving anew problem.
- Often people, especially novices, are not reminded of the most appropriate sets of cases when they are reasoning.
- Case data can be hard to gather.
- Predictions are limited to the cases that have been observed

2.3.8 CBR and Other Reasoning Method

Other than case-based reasoning, they are many other reasoning methods ca be used in solving problems. Rule-based reasoning and model-based reasoning are both the most common reasoning method to be used in this field. How does case-based reasoning compare with other heuristic methods?

The major difference between case-based and rule-based reasoning is the size of the chunks used for reasoning. This leads to a number of other differences, most of which is that rule-based reasoning is a process of composing large number of small chunks to get

a solution, while case-based reasoning is a process of adapting small of large chunks. Case-based reasoning been shown to be more efficient in several solutions.

The relationship between case-based and model-based reasoning is more complementary. Both were created to avoid reasoning from scratch, and both are committed to reasoning with large chunks of knowledge. The knowledge they use, however, is quite different, with models representing general knowledge and cases representing specific knowledge. The conclusion is that both are needed for reasoning about complex, real-world situations, especially commonsense in which much is unknown.

2.4 TRADITIONAL CHINESE MEDICINE

Chinese Herbal Medicine is the oldest practiced medicine which goes back more than 5,000 years. The discovery of herbal medicine is ascribed to the legendary emperor Shen Nong (3494 BC). He introduced agriculture to his people and became enchanted by the medicinal properties of various plants. The discoveries of the Shen Nong era were passed from generation to generation by word of mouth, since there were no written records at this time. Myths and facts are therefore hard to distinguish.

It took approximately 2,000 years until the discoveries of Shen Nong and his followers were committed to writing. Many remedies and concepts could stand the test of time and their effectiveness has been proven by modern science as well.

Chinese medicine was given an academic boost in the West in 1931: the wealthy American businessman **G. M. Gest** was cured from an eye disease by a Chinese physician after all other efforts had failed. He was so grateful that he collected 75,000 books about Chinese medicine and established the Gest Oriental Library at Princeton University. Much of the science of Chinese herbal medicine has been confirmed by Western research since then.

2.4.1 Chinese Medicine and Western Medicine

Chinese and Western medicine uses different viewpoints and methods to conduct research and explore the science of human life patterns. In comparison with Western medicine and modern science, the basic characteristics of TCM theory are abstract and indistinct. However, the continuous process from specific to abstract and abstract to actual in the formation and development of the theory of TCM is the work of countless doctors over many thousands years. The conception and development of the basic theory of TCM is similar to that of other natural sciences. It is concurrent with the philosophy of that era. At the core of TCM is the theory of "ZangXiang" which is built on the theories of the Five Elements and "Yin and Yang" which originates from ancient Chinese philosophy. As such, the theories of TCM, the Five Elements and "Yin and Yang" are inseparable.

Life is an extraordinary complicated system that is unbounded. In the study of the life sciences, TCM and Western medicine have different views and different methods of research,

Therefore their theories are different. In western medicine, modern scientific methods were used to make a detailed study of the structure of the human body. But Man and his body are infinitely different concepts. The human body is made up of elements, cells, tissues, organs and systems. On this foundation, Man builds various abilities such as thought and will-power. In other words, the word "Man" becomes more meaningful, bringing us closer to the true meaning of human life. From a certain point of view, TCM emphasizes more on Man while western medicine focuses more on the human body.

2.4.2 Traditional Chinese Herbs

Have you ever wondered why is it that the Chinese live healthier and longer lives than any other people in the world? One of the main reasons is their use of rejuvenating herbs. It's no secret then, that herbs help improve the quality of our health.

Therefore, the Chinese have had such remarkable success with herbal curative formulas and they have been developing, improving and perfecting them for 5000 years.

The Chinese believe that proper diet and exercise, meditation and the careful prescription of herbs can prevent disease, restore the body's proper balance and aid one in maintaining optimum health.

Chinese herbology posits that all disease stems from imbalance occurring in the organ system. The healthy body, with its properly functioning immune system, is more than capable of combating the viruses it is exposed to. A properly balanced system will resist and destroy bacteria before they cause disease.

Simply put, illness simply cannot exist if the body is in harmony. On the other hand, Chinese herbal theory holds that an unhealthy body requires herbal and dietary treatments to restore and maintain this essential balance.

2.4.2.1 *Introduction to Traditional Chinese Herbs*

Chinese herbs are a major component of Chinese medicine. Traditional Chinese medicine is based on the belief that the same forces that govern the natural world affect health and illness. Chinese herbs are prescribed for specific conditions and symptoms, as well as to maintain health and vitality.

There are more than 2,000 Chinese herbal treatments, although only a few hundred are in general use. They are powerful, effective and safe when used properly—in the right dose and in the proper combinations. Most Chinese herbs are derived from plants and may include the roots, seeds, bark, leaves and branches. However, minerals and animal substances may also be used, often in combination with plant-based herbs. Herbs are carefully processed to remove toxins and debris. Today most Chinese herbs are given as formulas, or mixtures of herbs, and are available in tablets or capsules, as well as in raw form.

Today increasing numbers of people seek out Chinese herbs because they are effective, less expensive than pharmaceutical drugs and are considered to be less toxic. According to Chinese clinical studies, these herbs, and others that have been added to the list of useful items over the centuries, can greatly increase the effectiveness of modern drug treatments, reduce their side-effects, and sometimes replace them completely.

In China, the two most common methods of applying herb therapies are to make a decoction (a strong tea that must be simmered for about an hour or more) and to make large honey-bound pills. Both of these forms meet with considerable resistance in Western countries. The teas are deemed too time-consuming, smelly, and awful-tasting to justify their use, and the honey pills (boluses) are sticky, difficult to chew, and bad tasting. Thus, modern forms that are more acceptable have been developed for most applications.

The herb materials used in all these preparations are gathered from wild supplies or cultivated, usually in China (some come from India, the Mid-East, or elsewhere). There are an estimated 6,000 species in use, including nearly 1,000 materials derived from animal sources and over 100 minerals, all of them categorized under the general heading "herbs." Herbs are processed in various ways, such as cleaning, soaking, slicing, and drying, according to the methods that have been reported to be most useful. These materials are then combined in a formulation; the ingredients and amounts of each item depend on the nature of the condition to be treated.

In some cases, a practitioner of Chinese medicine will design a specific formulation for an individual patient, which might be changed frequently over a course of treatment. In

other cases, one or more formulas already prepared for ingestion without modification are selected for use. The outcome is monitored, and the determination of whether to continue the current formula, change to another, or discontinue use is made on the basis of actual versus desired outcomes and the obvious or subtle effects of using the herbs.

The main reason that more Westerners are turning to Chinese herbs rather than local herbs is because of the vast scope of experience in using the Chinese materials. In every province of China, there are large schools of traditional Chinese medicine, research institutes, and teaching hospitals, where thousands of practitioners each year gain training in the use of herbs. The written heritage of Chinese medicine is quite rich. Ancient books are retained, with increasing numbers of commentaries. New books are written by practitioners who have had several decades of personal experience or by compilers who scan the vast diverse modern literature and arrange the results of clinical trials into neat categories.

Adverse responses to Chinese herbs are monitored at the Institute for Traditional Medicine through its contacts with numerous practitioners around the country and subscriptions to technical journals published in China and Japan. Negative interactions with Western drugs have not been noted for any of the common herb materials when used in the normal dosage range. A few people experience allergic reaction to individual herbs, a problem that often cannot be predicted in advance since these are idiosyncratic responses. A more common reaction is a gastro-intestinal response, which might include constipation or diarrhea, nausea or bloating. Such reactions may occur if the individual has poor digestive functions, or if the herbal formula is not quite right for the needs of the individual. Taking the herbs at a different time in relation to meals may be helpful in

resolving some of the gastro-intestinal reactions. In a few cases, use of Chinese herb formulas may cause dizziness, headache, agitation, sleepiness, hungry feeling, lowered appetite, sensation of heat or cold, or other sensory reactions. If such responses persist after about three days of using the herbs, it may be necessary to change formulas.

2.4.2.2 *Conditions not appropriate*

Chinese herbs are effective for most conditions in the proper combination. However, in some conditions, such as pregnancy, there are many herbs that should be avoided. Herbalist can determine the best combination for patient's condition after a full consult and examination.

2.4.2.3 *Herbs combining to make an effective treatment*

An ancient formula prescribed for the initial stage of an infectious disease is Cinnamon Combination. It includes cinnamon, peony, licorice, and ginger. It is said that the cinnamon (twig) and peony coordinate the circulation at the surface of the body (where disease is believed to enter) and relaxes tense muscles. Ginger and licorice improve the digestive functions and improve the body's healing energy. An ancient formula used to treat chronic illness is Ginseng and Tang-kuei Ten Combination. It includes astragalus, ginseng, atractylodes, hoelen, licorice, cinnamon, tang-kuei, peony, and rehmannia. Astragalus, ginseng, atractylodes, hoelen, and licorice promote digestive functions,

increase the energy, nourish the internal organs, and enhance weakened immune responses. Cinnamon (bark) warms up the weakened metabolism. Tang-kuei, peony, and rehmannia nourish the blood. Another ancient formula, used for a variety of diseases and function disorders, is Minor Bupleurum Combination. It includes bupleurum, ginseng, ginger, hoelen, and licorice. Bupleurum harmonizes the circulation between the internal organs and the body surface, it alleviates stress in the chest and abdomen, and it reduces inflammation. As indicated above, ginseng, ginger, hoelen, and licorice benefit the digestive processes and increase energy.

All of these formulas are widely used today, often by making some slight modifications to address the particular needs of the individual or the characteristics of the disease. For example, Cinnamon Combination (with appropriate modifications) has been used in Chinese clinical trials for treatment of frostbite, pernicious vomiting of pregnancy, and appendicitis. Ginseng and Tang-kuei Ten Combinations has been applied to treatment of side-effects of cancer therapy and for prevention of cancer recurrence after successful treatment. Minor Bupleurum Combination is one of the formulas frequently given in cases of chronic hepatitis B infection, and it is also used for inflammation of the stomach and pancreas.

2.5 CASE-BASED REASONING DEVELOPMENT TOOLS

To build a CBR system, a shell or development tools will be needed. The CBR developer can either build their own or buy the available tools from the market.

A CBR system tool, or shell, is a software development environment containing the basic components of CBR systems. Associated with a shell is a prescribed method for building applications by configuring and instantiating these components.

Despite the fact that CBR is a recent innovation there are at least ten commercially available tools with CBR functionality. This is an impartial review and has no allegiance to any tool vendor.

2.5.1 A Theoretically Ideal Feature Set

A theoretically ideal CBR tool should support each of the main processes of CBR (i.e., retrieval, reuse, revision and retention). In addition, the tool should support the developers in delivering an efficient solution and the tool must integrate with other systems. The following are the recommended function set:

- ***Representation*** - this must support a full range of data types (e.g., numeric, string, Boolean and symbol) and should be able to structure cases in ways relevant to the application domain. Flat records of value: attribute pairs may be sufficient for some applications, but complex domains may require ordered

symbol hierarchies, relationships between features and may benefit from object-oriented inheritance.

- ***Retention*** - the case-base should be organized into a manageable structure that supports efficient search and retrieval methods. A balance should be found between storing methods that preserve the semantic richness of cases and their indices and methods that simplify the access and retrieval of relevant cases.
- ***Retrieval*** - indexing of case will be necessary to make retrieval efficient so indexing must be supported. It may be automatic, but developers should be able to influence the process. If nearest neighbor is used then case features should be able to be weighted and similarity measures customized. If inductive techniques are used the index tree generated should be open to inspection and alteration by developers.
- ***Revision*** - this requires the provision within the tool of a programming language for case adaptation. The language may be procedural or use KBS techniques.

The following are some other issues that are relevance to developers:

- The tool should be able to manage large case-bases with retrieval times increasing linearly (at worst) with the number of cases.
- The tool should support a variety of retrieval mechanisms and allow them to be mixed when necessary.
- The tool should provide the developer with a variety of metrics to both assist the development of an efficient system and the maintenance of the case-base.
- Since organizations may hold case data in existing databases the tool should be able to import data from the full range of corporate databases.

- A CBR tool should provide a good user interface both for the developers and for operational users.
- Since it may be necessary to embed the developed application the tool should provide a C function library (or similar e.g., a DLL) or support the use of communication protocols such as MS Windows Dynamic Data Exchange (DDE).

2.5.2 CBR Software Tools

Theoreticians might argue that the current surge in interest in CBR is due to the intuitive nature of CBR and because it may closely resemble human reasoning. Software vendors might argue that it is because CBR tools have made the theory practically feasible. There is truth in both views but certainly the tools have made a contribution. This section reviews most of the currently available major CBR tools. The tools are dealt with in alphabetical order. The section concludes with a table that summarizes the functionality of the tools reviewed.

Table 2.2: Summary of CBR software tools

Product	Platforms	Representation	Retrieval	Adaptation	Interface
ART*Enterprise Inference Corp.	a wide variety of PC, workstation, DEC, IBM HP	Flat value attribute pairs supporting a full range of variable types	Nearest neighbor but can be augmented	Functions, rules and other knowledge-based	fully featured GUI builder

	and mainframe environment		using ART's programming environment	techniques.	
CASE-1 Astea International	PC Windows	Flat records supporting text and weighted questions	Nearest neighbor and knowledge-guide retrieval	no adaptation features	Interface cannot be customised
CasePower Inductive Solution Inc.	PC Windows, Macs OS/2	MS Excel Spreadsheet ordered symbol hierarchies and nested cases	Nearest neighbor	via Excel functions and macros	Excel interface
CBR2 Inference Corp.	PC Windows and MVS version	Flat records supporting text and weighted questions	Nearest neighbor and knowledge-guide retrieval	no adaptation features	Tool Book Interface of CBR-Express can be customized
Eclipse The Halley Enterprise	any ANSII C environment	Flat value attribute pairs supporting a full range of variable types	Nearest neighbor	Functions, rules and other knowledge-based techniques.	no interface it is only supplied as C library
ESTEEM Esteem Software Inc.	PC Windows UNIX/X Motif	cases can be order hierarchically and can be nested	Nearest neighbor and inductive (ID3) 1	functions and rules	GUI builder

			retrieval		
KATE *AcknoSoft*	PC Windows UNIX	Hierarchical cases	c		Tool Book Interface can be customized
ReCall Isoft	PC Windows UNIX	Hierarchical cases with relationships	Nearest neighbor and inductive	deamons	Graphical development environment
ReMind Cognitive System Inc.		Object Oriented Hierarchical cases	Hierarchical cases	rules	Customizable interface

From the various type of CBR shell mentioned above, I have chosen Art*Enterprise as my CBR system development tool:

ART*Enterprise

Many of the tools now offer some data integration but ART*Enterprise offers the best data integration along with excellent adaptation facilities through its powerful knowledge representation and programming environment.

ART*Enterprise is the latest incarnation of Inference Corporation's flagship development product. Inference Corporation based in California is one of the oldest established vendors of AI tools. Inference market ART*Enterprise as an integrated,

object-oriented applications development tool is designed for MIS developers and offering a variety of representational paradigms including:

- a procedural programming language
- objects supporting multiple inheritance, encapsulation and polymorphism
- Rules and cases.

This is all packaged with a GUI builder, version control facilities, and an impressive ability to link to data repositories in most proprietary DBMS formats for developing client-server applications. Moreover, ART*Enterprise offers cross-platform support for most operating systems, windowing systems and hardware platforms.

The CBR component in ART**Enterprise* is essentially the same as that in CBR Express (or rather vice-versa since CBR Express uses code from ART to provide its CBR functionality). However, because developers have direct access to the CBR functionality ART**Enterprise* is more controllable than in CBR Express.

In conclusion, ART*Enterprise is perhaps the ideal tool for embedding CBR functionality within a corporate wide information system. Although the CBR functionality itself is more limited than some tools (i.e., cases are flat value: attribute pairs and there is no support for inductive indexing) the proven knowledge representational abilities of ART will make it a good tool for performing complex case adaptation. It can be assumed that since ART*Enterprise uses similar code to CBR Express that its case retrieval times will be as fast (or faster) than those recorded in Althoff et al's experiments [95].

2.6 REVIEW OF EXISTING COMMERCIAL SYSTEM

The applications of CBR in the medical field are mostly for academic research and not as commercial product. The following are some of the commercial CBR systems which are mostly in the field of engineering.

2.6.1 CaseLine

CaseLine is a first generation technology demonstrator used by British Airways (BA) to assess the potential of CBR. CaseLine assists Boeing 747-400 technical support engineers with aircraft fault diagnosis and repair between aircraft arrival and departure. It advises on past defects and known successful recovery and repair procedures. When a fault in a Boeing 747-400 is detected or suspected, either by monitoring equipment or the pilots, details are transmitted to ground staff. The plane may only be scheduled to be on the runway for one hour during which time engineers have to identify the cause of the fault and effect repair. This is complicated because defects are often obscure and have complex and inconsistent causes. To delay the plane will disrupt schedules and costs thousands or pounds per minute. To let the plane take off with an unresolved fault could have catastrophic consequences

CaseLine is implemented in ***ReMind*** (See Table 2.2). Users can input diagnostic information and control the search for available repair and recovery information. The system contains around 200 cases that describe previous failure instances and details of successful recovery actions.

Three main search modes are provided:

- ***ATA Chapter*** - a simple two digit number referring to a fault in the plane's maintenance manual
- ***EICAS Message*** - a precise but variable length alphanumeric text indicating a fault
- ***Reported Defect*** - a variable length string describing a fault.

These can be used alone or together for case retrieval using either nearest neighbor or inductive retrieval. Usually, a single case is retrieved if an exact match has been specified or several cases if a partial match was required. CaseLine helps engineers identify procedures that have the highest likelihood of success. The used of CaseLine has reduced costly delays by cutting out less productive routes to fault analysis and fault finding.

Three benefits of CBR identified by BA are:

- Intuitive to both developers and users
- Complementation of human reasoning and problem solving
- Retaining the rich context of a problem situation - they discard nothing but simply index on different features of what they store

The latter point is of particular legal interest. If a rule-based diagnostic system were developed its rules would represent distillated knowledge. The original reasons why a rule was created may become obscured with time. However, episodic cases are always heavily contextualized. If BA were sued for negligence using CaseLine they could

demonstrate that engineers had followed procedures that had proved successful in case X with a simple defense. However, if BA used a rule-based system expert witnesses might have to prove that a rule was *theoretically* correct, which is a more complex defense.

2.6.2 CLAVIER

CLAVIER is a CBR system developed by Lockheed Company. Lockheed is an aerospace company, which produce elements for modern aircraft that are made up from composite materials. These materials require curing in large autoclaves. Each part has its own heating characteristics and must be cured correctly. If curing is not correct the part will have to be discarded. Problems occurred when the autoclave's heating characteristics are not fully understood. This is complicated by the fact that many parts are fired together in a single large autoclave and the parts interact to alter the heating and cooling characteristics of the autoclave.

Operators of Lockheed's autoclaves relied upon drawings of previous successful parts layouts to inform how to layout the autoclave but this was complicated by the fact that layouts were never identical because parts were required at different times and because the design of the composite materials was constantly changing. Consequently operators had to select a successful layout they thought closely matched and adapt it to the current situation.

Lockheed decided to implement a CBR knowledge-based system to assist the autoclave operators because it is closely resembled the CBR paradigm. Their objectives were to:

- Reuse previously successful loadings
- Reduce the pressure of work on one or two experts
- Secure the expertise of the experts as a corporate asset
- Help to train new personnel

The development of CLAVIER started in 1987, and it has been in regular use since the autumn of 1990. CLAVIER searches a library of previously successful autoclave layouts. Each layout is described in terms of:

- Parts and its relative positions on a table
- Tables, and its relative positions in the autoclave
- Production statistics such as start and finish times, pressure and temperature.

CLAVIER finds substitutes for parts in a layout that do not match, and it recommends new layouts to operators. In adapting new layouts from previous ones CLAVIER:

- Creates new layouts by adapting pieces of previous layouts
- Minimizes the number of required parts not included in the layout
- Maximizes the number of high-priority parts included in the layout
- Maximizes the total number of parts in a layout.

CLAVIER acts as a collective memory for Lockheed and as such provides a uniquely useful way of transferring expertise between autoclave operatives. In particular the use of CBR made the initial knowledge acquisition for the system easier. Indeed, it is

doubtful if it would have been possible to develop a MBR system since operatives could not say why a particular autoclave layout was successful. CLAVIER also demonstrates the ability of CBR systems to learn. The system has grown from 20 to over 150 successful layouts and its performance has improved such that it now retrieves or adapts a successful autoclave layout 90% of the time.

CHAPTER 3: METHODOLOGY

This chapter will describe the methodology that I had used to complete my application system and the description of system analysis.

3.1 CONCEPT OF METHODOLOGY

Methodologies provide comprehensive guidelines to follow for completing every activity in the system development life cycle (SDLC), including specific models, tools and techniques. A methodology might be homegrown or purchased from a consulting firm.

3.1.1 Waterfall Model

I have chosen this model is because it really provides a high-level view for me to develop this project, this model was chosen because of following reason:

- **Very structured**

  The system is design using a logical flow.

- **Predictable**

It allows estimation of the completion of each stage so that the system can be developed within the time frame given.

- **Involves user participation**

  Require information gathering form the user in order to develop a system that meet user needs to a greater extend.

- **Good visibility**

  All the requirements can be identified and well defined.

- **More efficiency**

  The time and resources can be well determined in order to enable developers to manage the project more efficiency.

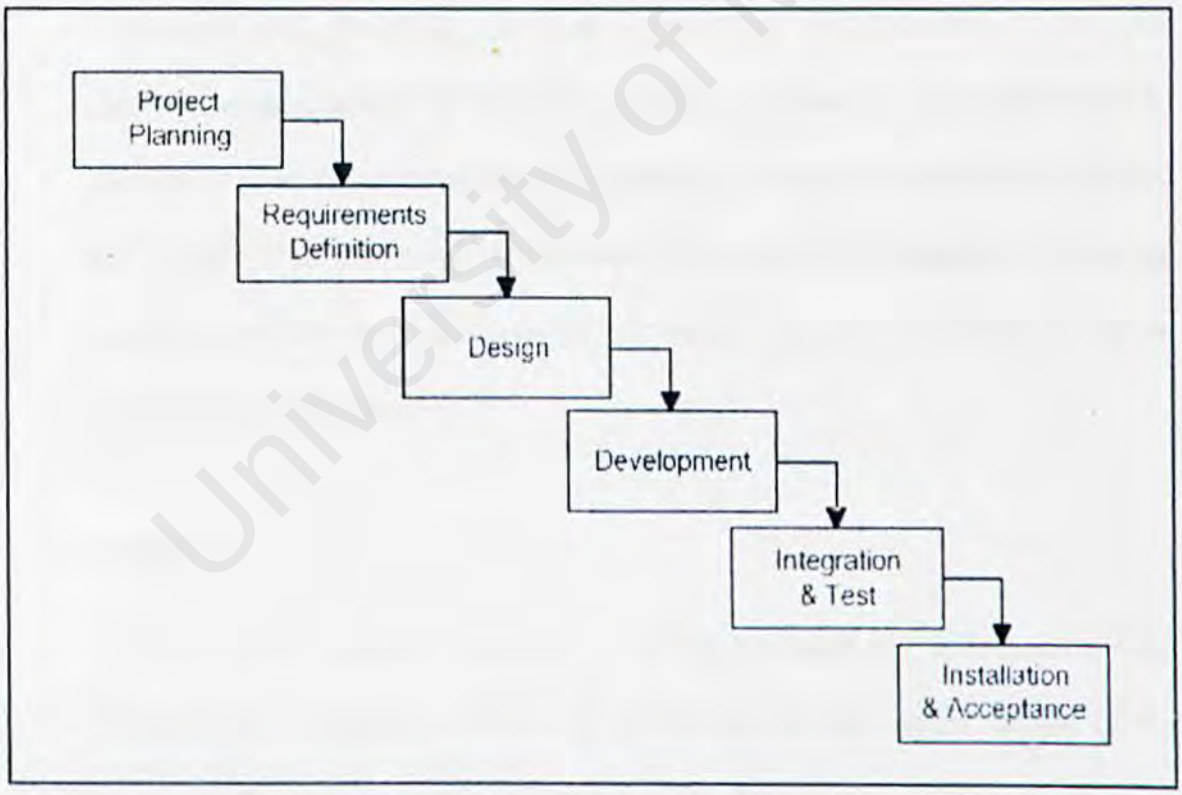


Figure 3.1: Waterfall Model

The following are the descriptions of each process in Waterfall Model that will be involved in this project:

a) Planning

A program management plan is developed that documents the approach to be used and includes the discussion of methods, tools, tasks, resources, project schedules and user input.

b) Requirement

In the first phase of the system development life cycle, the system analysis is concerned with identifying problems, opportunity and objectives. The real problem is determined and the best solution is decided. Opportunities can be conceived of as the observation of the problem, and improvement can be defined as changes that will result in increment yet worthwhile benefits. Then the specification and the constraints of the project can be determined to define information requirement.

c) Design

In this stage, planning is work out so that to meet the specification. The information collected earlier is use to accomplish the logical design of the system. The analyst will design accurate data-entry procedures, files or databases, user output (either on-screen of printed) that meets information needs,

and finally the controls and backup procedures to protect that system and the data.

d) Coding

During this phase, the analyst works with the programmers to develop any original software that is needed. Programmers have a key role in this phase because they design, code and remove syntactical errors form the program. To ensure quality, a programmer may conduct either a design or a code walk-through to explain complex portions of the program to a team of other programmers.

e) Testing

Before the system can be used, it must be tested. It is much less costly to catch problems before the system is signed over to users. A series of tests to pinpoint problems is run first with sample data and eventually with actual data from the current system.

f) Operation

In this last phase of the system development, the analyst helps implement the system that involves training the users to handle the system. Some training is done by the vendors, but oversight of trainings is the responsibility of the system analyst. Additionally, the analyst needs to plan for a smooth conversion files form old systems to new one. This process includes converting files from old

formats to new ones or building a database, installing equipment and bringing the new system into production.

g) Documentation

This integrated process takes place every phase. Activities of each phases are documented in the report form so that to provide a clear view of the progress of the each stage.

3.2 CASE-BASED REASONING OVERVIEW

Case-Based Reasoning (CBR) is modeling human in solving problem by relying heavily on memory of past experiences, which is called cases in CBR. Consider a simple everyday problem such as using an unfamiliar machine. Rather than starting from scratch when learning how to use a new machine, people rely on memory of experiences with other machines. These memories allow them to derive shortcuts and anticipate problems that might arise by having previously observed and solved similar problems.

Although previous cases provide important information for dealing with the current situation, they are not exactly the same. It would be necessary for the person to have to provide a perfect description of the previous case in order to retrieve it from memory. People are very good at recognizing the similarities in situations while traditional programming techniques require exact matching.

CBR contains the following features:

- Description of the features of and solution to previous problems to assist in solving a current problem. These descriptions are called cases.
- A description of the features of the current problem. This description is called the presented case.
- A matching operation which provides partial rather than complete matching.

The matching operation matches a presented case with previous cases by accumulating pieces of evidence that the current case is similar to a stored case. A scoring formula converts the accumulated pieces of evidence into a single score for the match between the presented case and any previous case. The stored case with the highest score has the best match to the presented case. Generally, the cases with the highest scores are returned by the case-based reasoning mechanism.

In ART* Enterprise's case-based reasoning mechanism, cases are represented as objects. The features used by the matching mechanism are attributes of the case objects. These cases are referenced by a case-base which is also an object.

3.2.1 Case Base Building and Usage

A case based application has the following components:

- Cases that represent problems and their solutions.
- A case base where the cases are stored.
- A matching operation for matching a new presented case against the cases stored in the case base.

A typical usage of case-based reasoning follows the sequence of events shown below:

Build a case base

- Create a case base that will hold the pertinent case information
- Define the significant features that describe the cases
- Submit cases to the case base as objects, whose attributes represent the case features.
- Save the case-base index to a file

Query a case base

- Load the case-base index file into ART*Enterprise
- Specify the match parameters, such as how many cases should be retrieved and how precisely they should match
- Build a presented-case object that describes a situation you'd like to compare for matching against the stored cases of the case base
- Pass this object to ART*Enterprise's match functions, receiving back a list of the best-matching cases in the case base
- Refine the presented-case object if necessary, repeating the process from step 3 until sufficient information has been retrieved to solve the problem

3.3 CBR VS OTHER REASONING TECHNIQUE

This section discusses the problems associated with developing Model-Based Reasoning (MBR) and Rule-Based Reasoning (RBR) knowledge-based systems. It posits that CBR appears to offer solutions to many of these problems, and presents evidence from the literature to support these claims.

Case-Based Reasoning (CBR) combines a cognitive model describing how people use and reason from past experience with a technology for finding and presenting such experience. CBR provides a conceptual framework in which to store operator experience and to later provide that experience to other operators to facilitate the situation assessment and solution formulation processes of RPD. This is accomplished by providing a context in which the human operator can view the current state and recent activities of the system and easy access to previous experience.

3.3.1 CBR Vs Rule-Based Reasoning

RBR knowledge-based system typically provides only rule trace backs as an explanation of their activities, which even an experienced system operator, has difficulty interpreting. In contrast, a case-based system is capable of explaining its activities in the context of the case from which it was reasoning, thereby giving the operator much more useful information to guide situation assessment.

Many complex systems are managed in control periods which decompose long durations of system management into control scenarios, or sequences of activities (e.g. aircraft

flight plan or a shift plan in a process control plant). Such control scenarios can be viewed individually as experiences, or cases. Together, they form a case base which is the basis of an automated knowledge-based system.

3.2.2 CBR Vs Model-Based Reasoning

As in MBR knowledge-based system, the last thirty years many knowledge-based systems have been developed that have an explicit model of the problem domain in which they operate. In many such systems the model is implemented by rules, and perhaps more recently by objects. In second generation systems a deep underlying causal model exists that enables the system to reason from first principles in its application domain. There is little doubt that such MBR systems (whether they are deep or shallow) can be very successful. However, there are five major problems with this approach:

- Knowledge elicitation is difficult
- KBS can be very complex and can take many man years to develop,
- KBS are frequently slow,
- KBS are often poor at managing large volumes of information,
- Once developed they are difficult to maintain.

3.2.3 CBR Advantages over Other Reasoning Techniques

People solve problems by using their experience. It is no surprise that expert and experience derive from the same root. We posit that the KBS community was seduced by rules and neglected the truism that experts solve problems by applying their experience, whilst only novices attempt to solve problems by applying rules they have recently acquired. The application of experience to problem solving is the hallmark of CBR. Thus, CBR is proposed by some as a psychological theory of human cognition and one that provides a cognitive model of how people solve problems. It offers a paradigm that is claimed to be close to the way people solve problems and one that overcomes the brittleness of MBR and RBR systems.

Hence, there is a strong case for CBR since it has several potential advantages over MBR and RBR:

- CBR systems can be built without passing through the knowledge elicitation bottleneck since elicitation becomes a simpler task of acquiring past cases.
- CBR systems can be built where a model does not exist.
- Implementation becomes a simpler task of identifying relevant case features, and moreover a system can be rolled out with only a partial case-base. This removes one of the bug-bears of KBS - how to tell when a knowledge-base is complete.
- CBR systems can propose a solution quickly by avoiding the need to infer an answer from first principles each time.
- Individual or generalized cases can be used to provide explanation that are perhaps more satisfactory than explanations generated by chains of rules.

- CBR systems can learn by acquiring new cases making maintenance.
- Finally, by acquiring new episodic cases CBR systems can grow to reflect their organization's experience. If a rule-based KBS were delivered to six companies and used for six months, after that time each system would be identical, assuming no maintenance had taken place. If six identical CBR systems were used in a similar way after six months there could be six different systems as each could have acquired different episodic cases.

3.4 DEVELOPMENT TOOLS

3.4.1 Hardware Tools

The development of this system is under the following hardware constraints:

- Pentium 4 1.6 GHz
- CD-ROM 56X Max
- Floppy Drive
- 256 Mbytes of RAM

3.4.2 Software Tools

The development of this system is under the following software constraints:

- Art*Enterprise

- Windows 2000 Professional

3.4.2.1 *Art* Enterprise*

Art* Enterprise is chosen as the system development tool because it offers the best data integration along with excellent adaptation facilities through its powerful knowledge representation and programming environment among the other similar software in the market.

ART**Enterprise* is the latest incarnation of Brightware's flagship development product. Brightware were formerly a division of Inference Corporation, one of the oldest established vendors of AI tools and the major player in the Case-Based Reasoning tool market.

3.4.2.2 Windows 2000 Professional

Windows 2000 Professional is chosen as the system development platform because of quite a number of reasons. Windows 2000 Professional includes a selection of improvements that make the operating system easier to use and easier to maintain, having a significant impact on the operating costs. Windows 2000 Professional helps make desktop computing easier to use through faster processing and simpler deployment and management.

The advantages of windows 2000 Professional:

- Reliability
- Mobility
- Manageable
- Usability
- Improved, Familiar Interface
- Works with Current Network and PCs
- High Security
- Easier Desktop Management
- Easier Deployment

3.4.3 System Recommendation

- Any Processor with the speed faster than 200 MHz
- 64 Mbytes of RAM
- CD-ROM
- Floppy Drive
- Windows 95/98 or compatible Operating System

3.5 SYSTEM REQUIREMENT

The system requirement need to be drawn out before develop a system. A requirement is a feature of the system or a description of the system is capable of doing in order to fulfill the system purpose. There are two types of requirement, which is as followed:

- Functional requirement
- Non-functional requirement

3.5.1 Functional Requirement

- An authentication and authorized module to protect system database from non-authorized user to access the system; password and identity are required to access the system
- Be able to prescribe the appropriate traditional Chinese herbs for a given illness
- Be able to check for herbs interaction among herbs prescribed
- Be able to provide an alternative herbs prescription should there be an interaction
- Be able to provide adequate explanation as to how it arrived at a particular conclusion
- Be able to provide help to assist user in using the entire topic mentioned above.

3.5.2 Non-Functional Requirement

- User Friendly - The design of system and interface should be user friendly and easy understanding by al user. The design of all interfaces should confirm to the following criteria:
    - Consistent, in term of screen design and error message displayed
    - High degree of understandability and avoid memorization of event and command
- Maintainability - It is the degree which the system can be cost affectively made to perform its function in a possibly changing operating environment. The system are easy to modify and test in updating process to meet the new request, correcting errors, or more to a different computer system
- Reliability - The degree in which the system operate in a user acceptable manner when used in the environment for which it was design, which does not produce dangerous or costly failure when it is applied in a reasonable manner
- Efficiency - Implementation of the system correspond to the most cost effective computing resource utilization, what process hat can be called or accessed in an unlimited number of time to produce similar outcomes at a creditable speed
- Security - Only the system administrator should modify knowledge base and inference engine with classification from domain expert
- Modularity
- Accurate and robust

CHAPTER 4: SYSTEM DESIGN

This chapter will discuss about the design of the system. Each stage in the process has been disrupted into more details. Data Flow Diagram (DFD) will be used to describe the facets in the proposed system where each of the modules in the system has been drawn out. The details represented in this chapter will serve as a reference and important guidance for the system development phase as well as the system implementation and maintenance phase.

4.1 SYSTEM OVERVIEW

System design is the creative process of transforming the problem into a solution and the description of the solution.

System design is a very important factor in system development as it determines the success of the system. The system specification describes the features of the system, the components or elements of a system and their appearance to users.

In order to meet the requirements mentioned in previous chapter, the guidelines for my system design are listed as below:

- **Specify logical elements**
    - Detailed design specifications that describe the features of a CBR system: input, output, and process.

- **Meet user requirement**

Meet user needs stated in terms of:

- Performing appropriate process correctly.
- Providing accurate results.
- Using appropriate method of interaction.
- Providing overall reliability.

- **Easy to use**
    - Favorable human engineering.
    - Ergonomic design that is physically comfortable and contributes to user effectiveness and efficient.

4.2 SYSTEM ARCHITECTURE

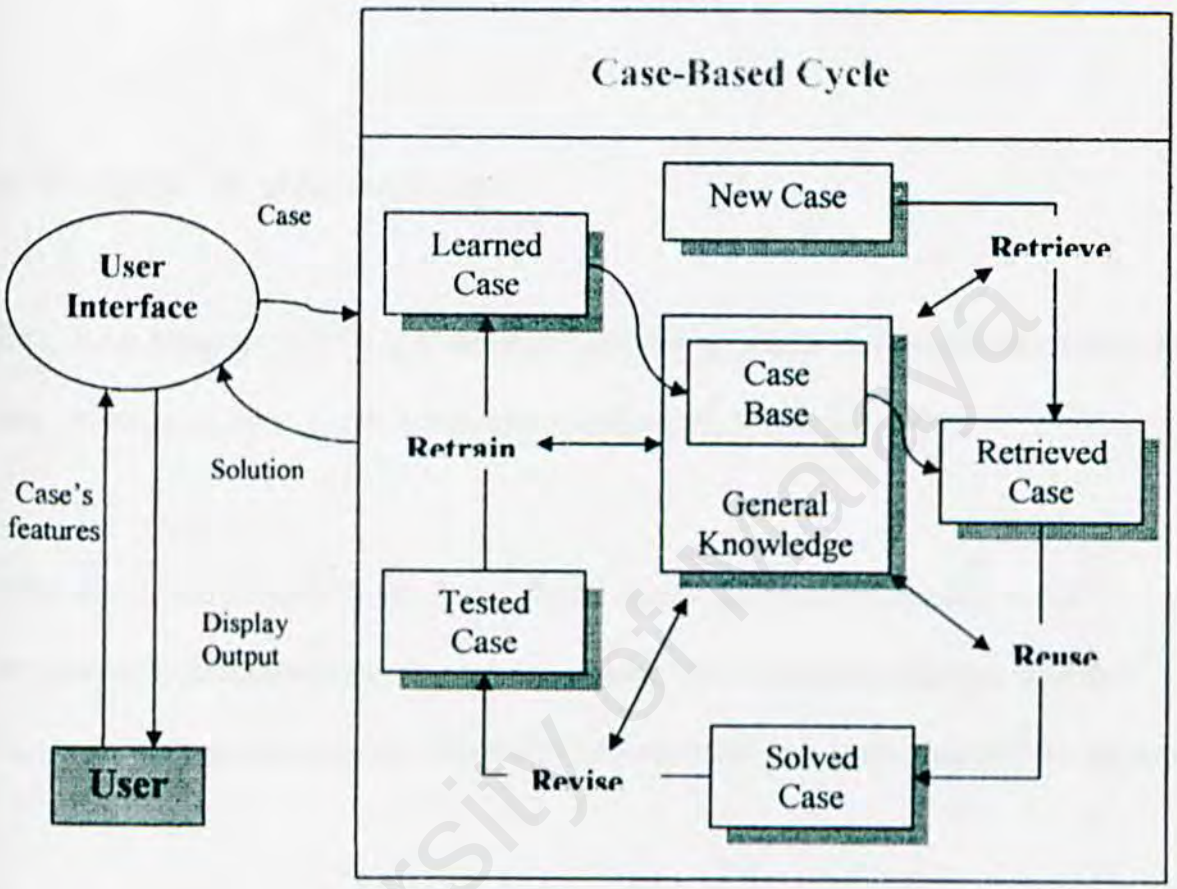


Figure 4.1: System Architecture

The following are the six major modules of this CBR system:

- Identify Module
- Match Module
- Reuse Module
- Adaptation Module
- Evaluation Module

- Retain Module

The system modules will be described by using Data Flow Diagram (DFD) along with the descriptions.

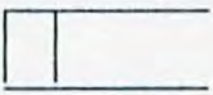
4.3 DATA FLOW DIAGRAM (DFD)

Data Flow Diagram (DFD) is a structure analysis approach that graphically represents data process and flows in a business system [Kendall & Kendall, 1999].

DFD depicts the broadest overview of system inputs, process and outputs, which correspond to data movement through the system. The diagrams crystallize how data moves within organization, the processes or transformation it undergoes and the outputs.

DFD is used in this project to represent the modules of the process involved and it is used in system design to form physical development. There are four major elements in DFD, which are entity, data flow, and process and data store. In this project, the symbols being used are following the notation of Gane and Sarson. The table 4.1 below summarizes the symbol.

Table 4.1: Symbols in Data Flow Diagrams (DFD)

Symbols	Meaning	Explanation
	Entity	A person, group, department or other system that can send data or receive data from the system
	Process	A transformation of data
	Data Store	A repository for data that allows addition and retrieval of data
	Data Flow	Movement of data from or to one process

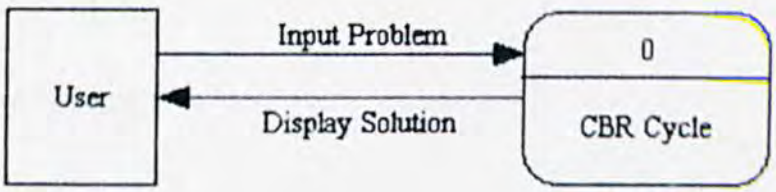


Figure 4.2 Context Diagram of CBR System

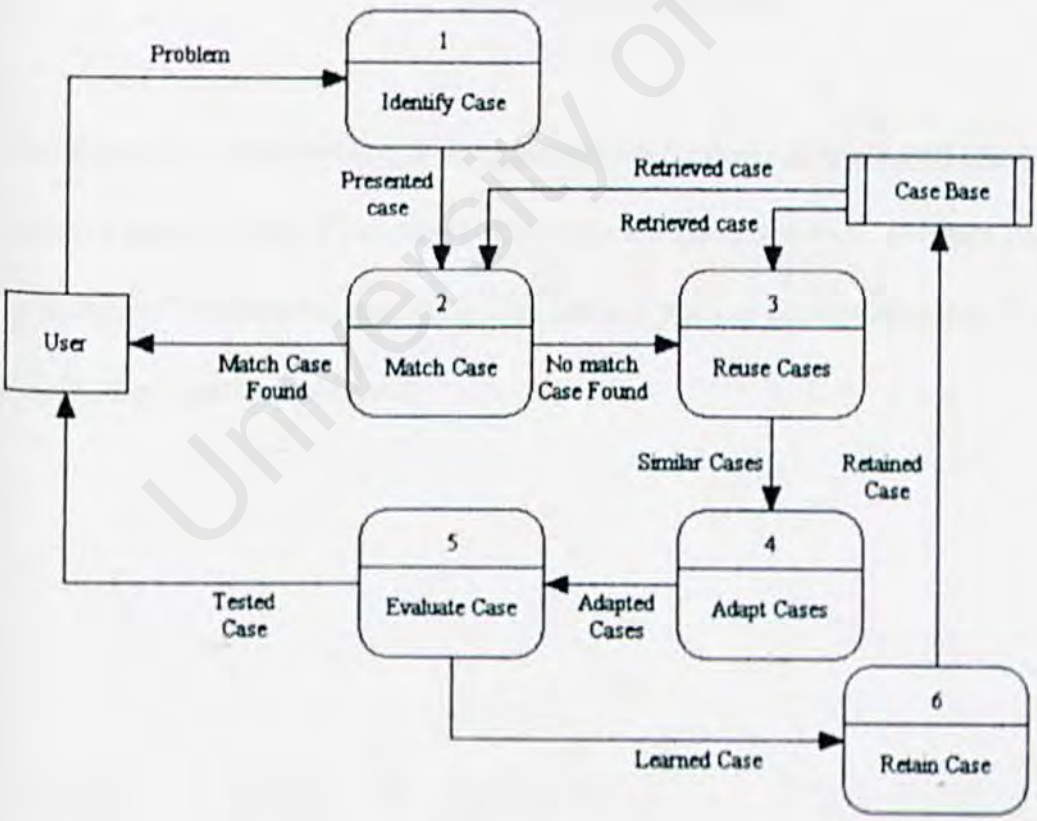


Figure 4.3: Data Flow Diagram of CBR System

4.3.1 Identify Module

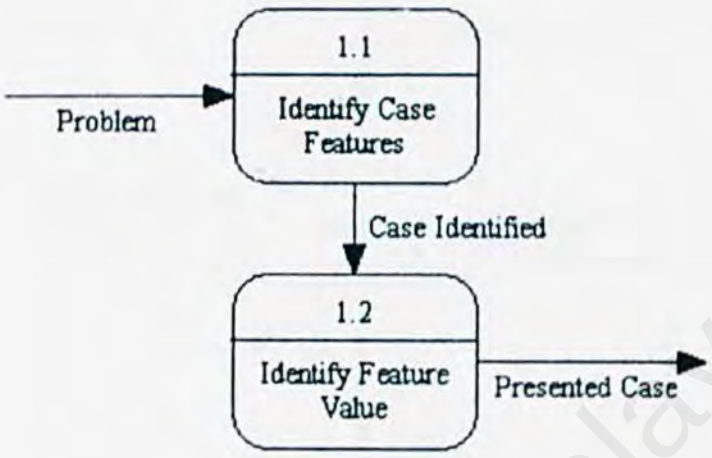


Figure4.4: Identify Module

The purpose of identify Module is to identify the features of presented case, which is the problem input by user. The system will index the presented case and then pass the indec to the Match Module for matching. The index is used as an identifier for the search of match case from the case base.

4.3.2 Match Modul

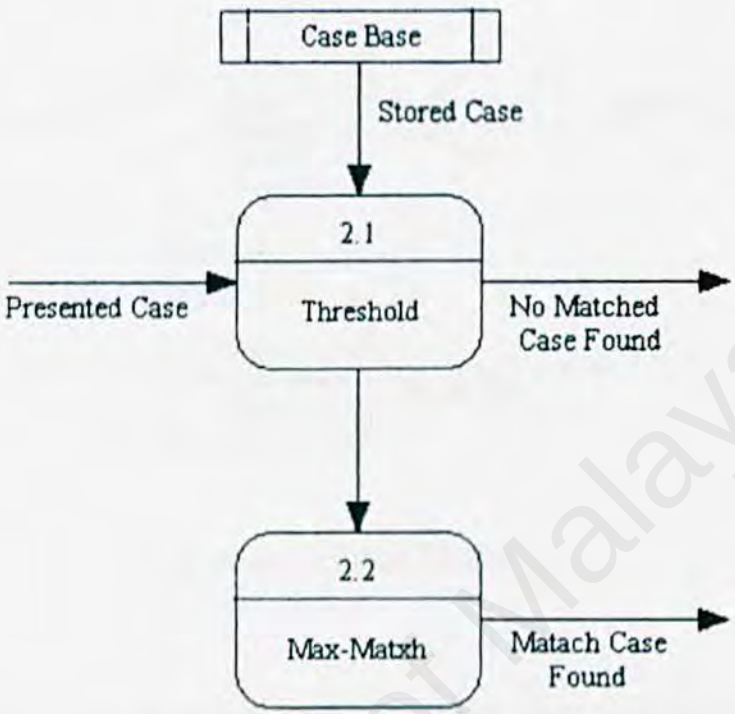


Figure4.5: Match Module

The system will try to retrieve a match case from the case base on the index given by the Identifier Module. When a match is performed on a large case base, only those cases that are of interest will be retrieve by properly set of threshold parameter and max-matches parameter. Only case that score greater than or equal to the threshold value are retrieved as match case while the max-match parameter specifies the maximum number of cases should be retrieved. Then the result will be display to the user as a solution for the presented case. When no exact matches are found, the index will be passing to the Reuse Module.

4.3.3 Reuse Module

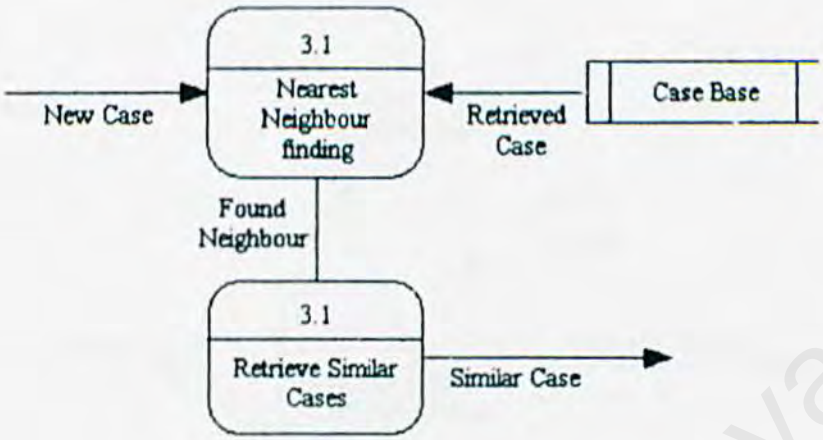


Figure4.6: Reuse Module

The Reuse Module will try to search for a few similar cases according to the similarity of the features of each case. Similarity of each feature depends on the feature's value but importance of different features may be different. Then these cases will be passing to the Adaptation Module.

4.3.4 Adaptation Module

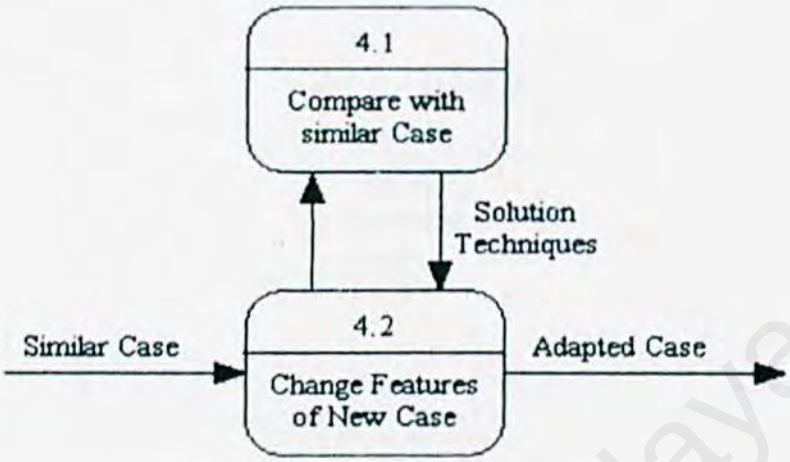


Figure4.7: Adaptation Module

The similar cases will have the similar solution techniques for the presented case but not exactly match. The cases will be adapted to the presented case by changing the features of the cases. Then the newly adapted cases will be passing to the Evaluation Module.

4.3.5 Evaluation Module

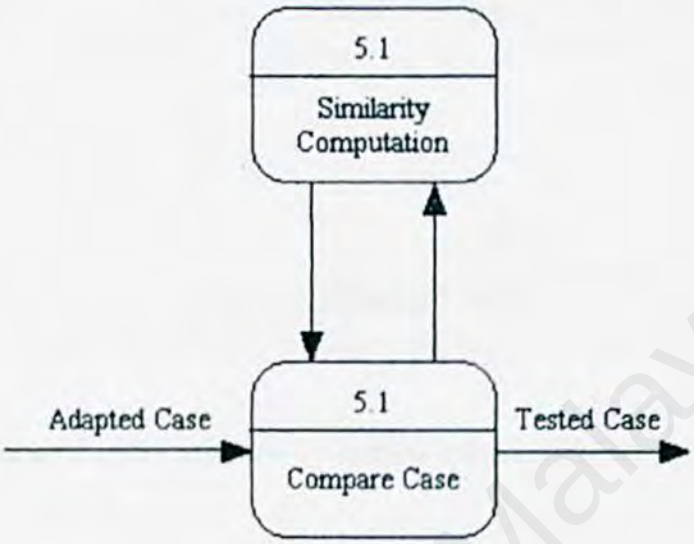


Figure4.8: Evaluation Module

The Evaluation Module will rank the cases according to their similarities. The system will access similarity base on similarity of each feature. The similarity will be compute by weight average and the most similar case will be display to the user as solution. If the diagnosis is correct, the new case will be sent to the Retain Module.

4.3.6 Retain Module

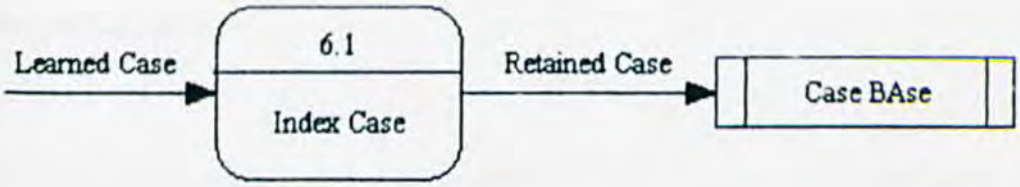


Figure4.9: Retain Module

The Retain Module will index the newly learned case and store the case into the case base.

4.4 CASE BASE DESIGN

In order to use ART*Enterprise's case-based reasoning facility, the system case base must be loaded. A case base is created by creating an instance of the object case-bases. In addition to creating the case base object, a case-base index must be created. This is a data structure which improves the efficiency of the case base matching operation.

The system will create a case-base index for case-base and stores it in memory. In order to save the case-base index for use at a later time, the case base index is written to a file.

At a later time, the case-base index can be read back into memory from the file and supplying the case-base index file name either with a file parameter or by storing it in the case-base object.

The information needed for matching cases is stored in a case-base index, an internal data structure that stores the relevant case and feature values needed to quickly match a presented case against the stored cases. The case-base index is stored in a file on a disk, and will be read into memory before cases and features can be modified or match operations can be performed.

Case in Case Base

A case is an ART*Enterprise object instance whose attributes constitute features of the case. Key attribute of a case is special. It has a value which is a unique integer between 0 and 65534. When the case is retrieved from a match operation, it is this value that is returned. If a case object has no key attribute or key attribute value, a warning is issued and a key attribute, with a unique value, is assigned by ART*Enterprise when the case is added to the case base.

Since the cases in a case base often have similar features, a class is defined in containing attributes for those features and makes the cases instances of that class. For example, for a disease case base, a disease class would be defined:

Once a case is defined, it must be added to the case base so that it is available to the matching mechanism. Before the case can be added to the case base, feature specifications must be added to the case base. If feature specifications are not added to the case base for a given feature, that feature will default to string matching with match and mismatch weights of 10 and -5. String matching requires that the value for a feature be a string and that it match exactly the value in the presented case. ART*Enterprise offers a much richer set of matching operations than string matching., The features will be defined before adding the case because once a case with a given feature has been added to the case base, it is no longer possible to redefine the feature specifications for that feature.

Name of Herbs: Cinnamomi_Rml***
Common name: cinnamon twigs (gui\ zhi-)
Dosage: 1-3 qian
Category: Relieving the Exterior
Positive matching

- **Main function :** RELIEVE WIND CHILL
- **Sub function :** Harmonizes the Ying (Nutritive function) and the Wei (protective) Qi

Negative matching

- **Caution :** Contraindicated in: Virulent Heat Evils
- **Toxicity :** Caused cardiac arrhythmias

Figure 4.10: Sample 1 of Case Design in Case Base

Name of Herbs: Crataegi_Fr***
Common name: hawthorn fruit (shan- zha-)
Dosage: 2-4 qian
Category: relieving Food Stagnation
Positive matching

- **Main function :** RELIEVE FOOD STAGNATION
- **Sub function :** Dissipates food accumulation

Negative matching

- **Caution:** Use cautiously in Deficiency of Spleen and stomach without food stagnation. Use cautiously in conditions with acid regurgitation.
- **Toxicity :** -

Figure 4.11: Sample 2 of Case Design in Case Base

4.5 SYSTEM USER INTERFACE

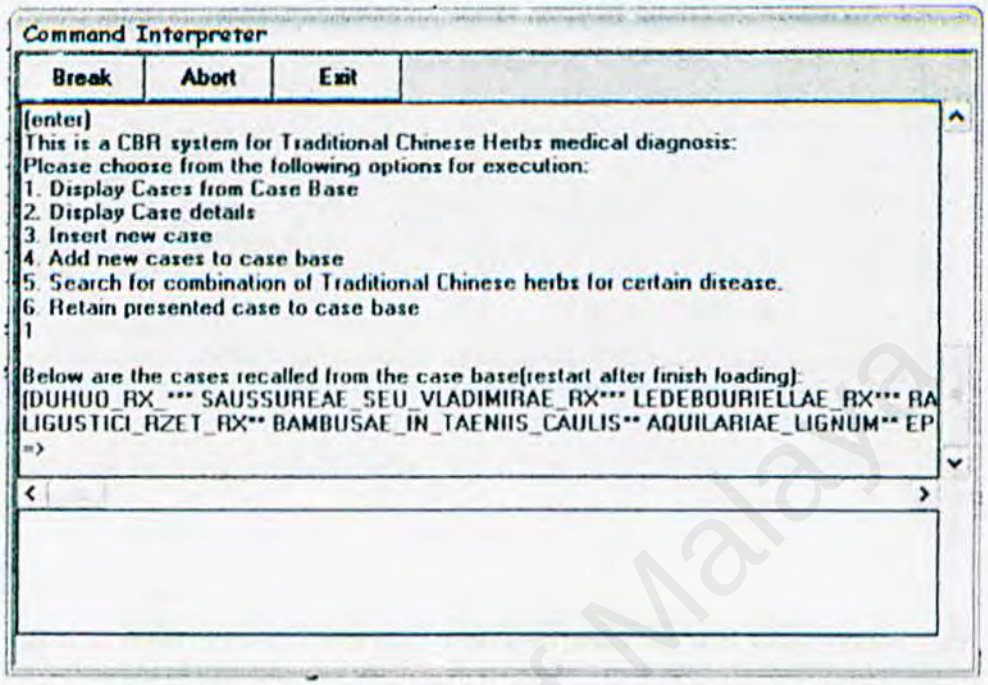


Figure 4.12: Recall cases from case base

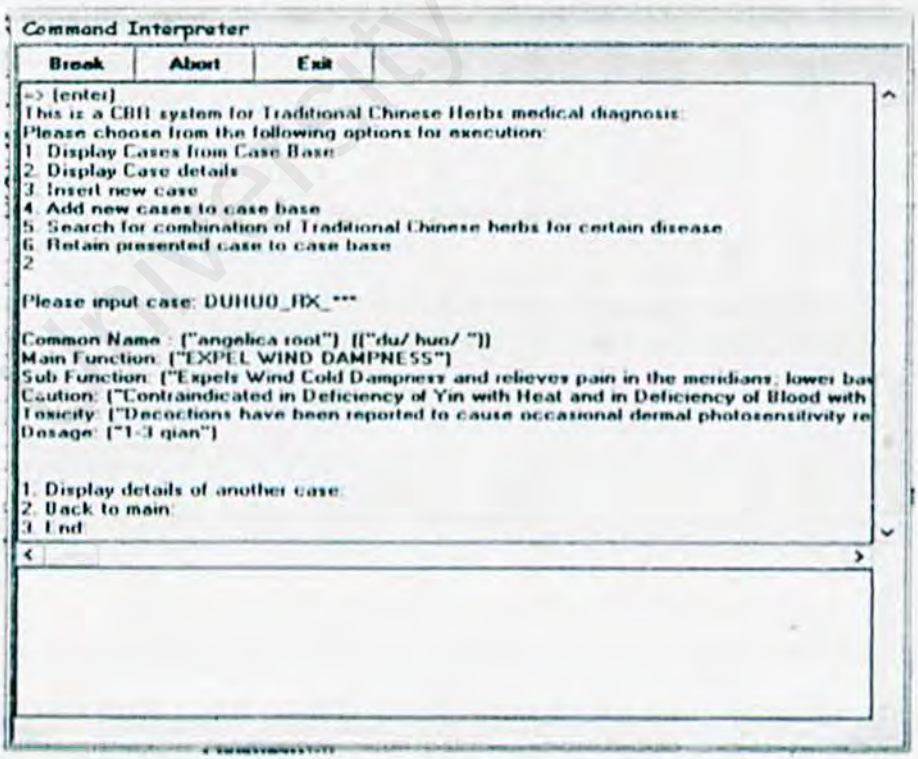


Figure 4.13: Recall case details

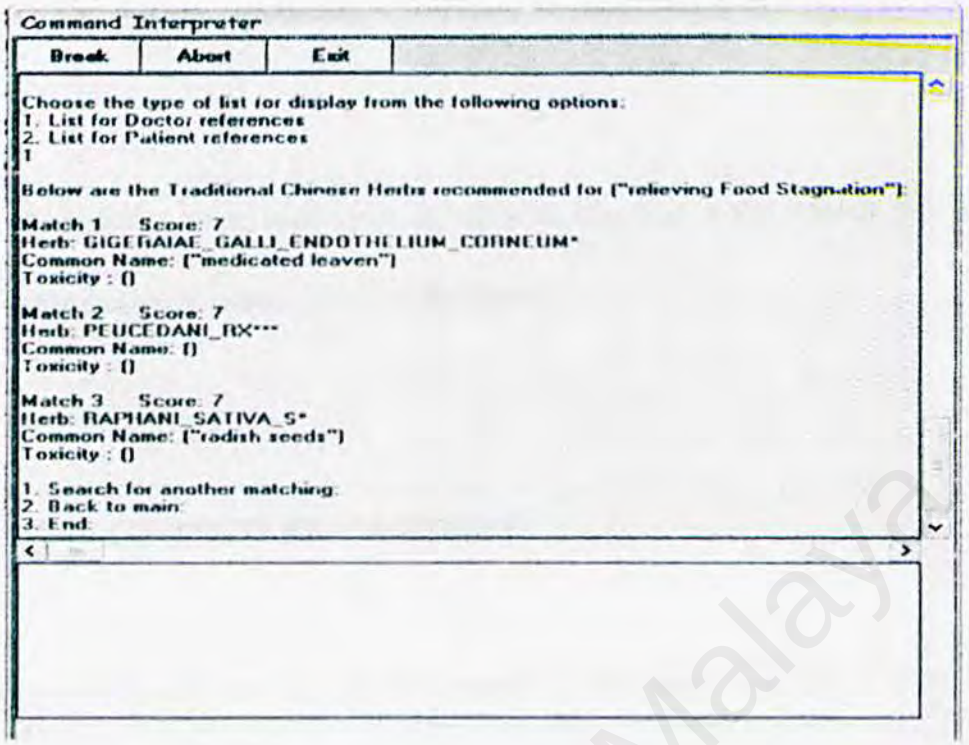


Figure 4.14: Case matching results for doctor references

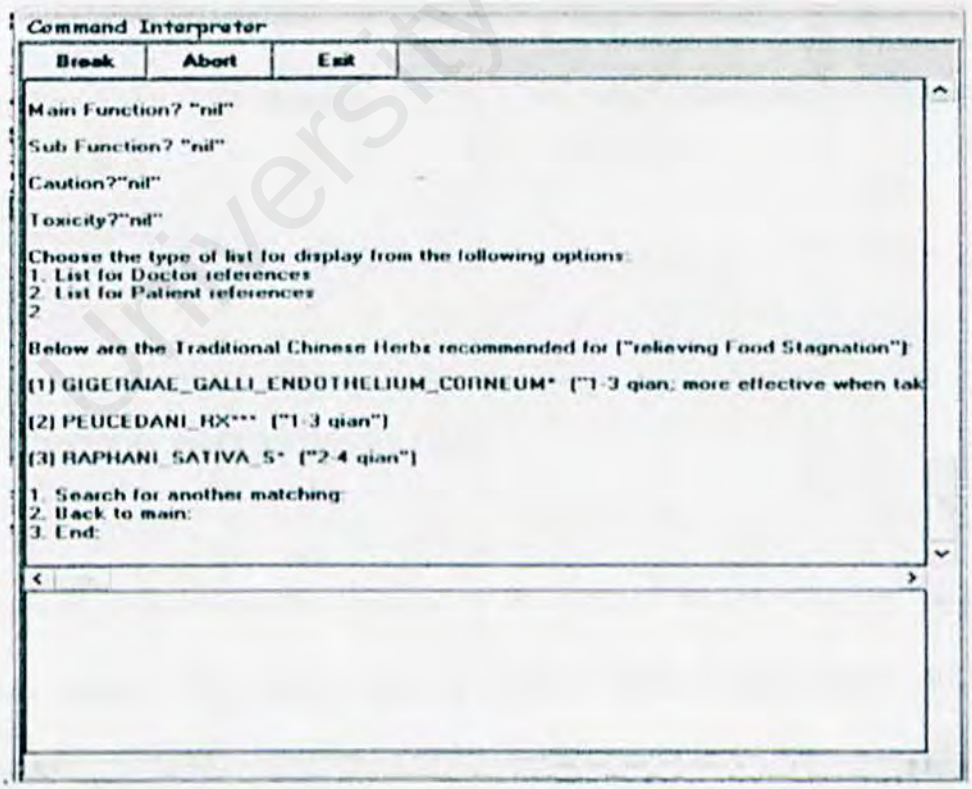


Figure 4.15: Case matching results for patient references

CHAPTER 5: SYSTEM IMPLEMENTATION

In this chapter, the steps and methods taken to implement the system that was design earlier in the previous chapter will be discussed.

5.1 DEVELOPMENT ENVIROMENT

The developing environment for the system is the tools used to develop the system which includes the hardware tools and the software tools.

Hardware Tools	Software Tools
• Pentium II Processor 233 MHz • 64 MB RAM • Ultra VGA 1024 Monitor • Mouse • Keyboard	• Windows 2000 proffesional • Art * Enterprise (Brightware, inc.)

5.1 SYSTEM IMPLEMENTATION

The system implementation include of the system design structure to a computer readable system. The system will be evolved from scratch design to a run able application. The following are the several implementations for this system.

5.2 FUNCTIONS IMPLEMENTATION

In this project, no graphic user interface has been implemented because the system will be focused on the application. A command Interpreter is used for data input and output which is very similar to MS Dos. The functions are created in order to allow user to select which operation to be executed from the option list and also to key in input for the CBR system. The functions will then generate outputs to be displayed to the user.

During the case matching process for this CBR system, user will just need to follow the instructions step by step and key in simple commands. Functions have made the system easier to use and understand instead of typing in the complicated commands. Below are the functions that have been created for the system:

```
(define-function enter ()
(printout t "This is a CBR system for Traditional Chinese Herbs medical diagnosis: " t)
(printout t "Please choose from the following options for execution: " t)
(printout t "1. Display Cases from Case Base " t)
(printout t "2. Display Case details " t)
(printout t "3. Add new cases to case base " t)
(printout t "4. Search for combination of Traditional Chinese herbs for certain disease" t)
(printout t "5. Retain presented case to case base " t)
(printout t "6. Insert new case" t)
(bind ?m (read))
   (if (= ?m 1) then (recall-case herb-case-base)
                     else (if (= ?m 2) then  (preprint-details)
                     else (if (= ?m 3) then  (preadd-case)
                     else (if (= ?m 4) then  (diagnose)
                     else (if (= ?m 5) then  (retain)
                     else
                     (if (= ?m 6) then  (insert)
                     else
                     (printout  t "Wrong input! Please try again..." t ))))))))
```

Figure 5.1: Start Up Functions

Matching New Case

```
(define-function check-nil (?m)
      (if(or(= ?m "NIL")
            (= ?m "nil"))    then
          (= ?m "")))
(define-global ?*main_function* = some-value)
(define-global ?*sub_function* = some-value)
(define-global ?*caution* = some-value)
(define-global ?*toxicity* = some-value)
(define-instance presented-case herb
   (main_function ?*main_function*)
   (sub_function ?*sub_function*)
   (caution ?*caution*)
   (caution ?*toxicity*))
(define-function diagnose ()
        (printout t "Main Function? ")
        (bind ?*main_function* (read))
        (printout t "Sub Function? ")
        (bind ?*sub_function* (read))
        (printout t "Caution?")
        (bind ?*caution* (read))
        (printout t "Toxicity?")
        (bind ?*toxicity* (read))
        (set-attribute-values presented-case
                main_function ?*main_function*
                sub_function ?*sub_function*
                caution ?*caution*
                caution ?*toxicity*)
        (check-nil ?*main_function*)
        (check-nil ?*sub_function*)
        (check-nil ?*caution*)
        (check-nil ?*toxicity*)
        (printout t "Choose the type of list for display from the following options: " t)
        (printout t "1. List for Doctor references " t)
        (printout t "2. List for Patient references " t)
        (bind ?m (read))
        (if (= ?m 1) then  (print-match herb-case-base presented-case)
        else      (if (= ?m 2) then (print-list herb-case-base presented-case)))
        (printout t "1. Search for another matching: " t)
        (printout t "2. Back to main:  " t)
        (bind ?m (read))
        (if (= ?m 1) then  (diagnose)
        else
         (if (= ?m 2) then  (enter)
        else              (printout t " Thank you for using this system!" t))))
```

Figure 5.2: Case matching Function

Print matched case with display properties for doctor reference:

```
(define-function presented-case (?presented)
(cbr:match-case herb-case-base ?presented)
(cbr:matches-found herb-case-base))
(define-function print-match (?case-base ?presented)
(presented-case ?presented)
(for ?m from 1 to 1 do
(printout t "Below are the Traditional Chinese Herbs recommended for "
(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-case ?case-base
?m) category collect$ ?value) ": " t t))
 (for ?m from 1 to (cbr:matches-found ?case-base) do
        (printout t "Match " ?m "    "      " Score: "
            (cbr:get-match-score ?case-base ?m)  t
"Herb: "(cbr:get-match-case ?case-base ?m) t
"Common Name: "(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-
match-case ?case-base ?m) common_name collect$ ?value)  t
"Toxicity : "(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-case
?case-base ?m) toxicity collect$ ?value)      t t)))

(print-match herb-case-base presented-case-1)
```

Print matched case with display properties patient reference:

```
(define-function print-list (?case-base ?presented)
(presented-case ?presented)
(for ?m from 1 to 1 do
(printout t "Below are the Traditional Chinese Herbs recommended for "
(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-case ?case-base
?m) category collect$ ?value) ": " t t))
(for ?m from 1 to (cbr:matches-found ?case-base) do
        (printout t "("?m ") "  (cbr:get-match-case ?case-base ?m) "  "

(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-case ?case-base
?m) dosage collect$ ?value) t  t)))

(print-list herb-case-base presented-case-1)
```

Figure 5.3: Matched Case Display Function for doctor and patient references

```
(define-function recall-case (?case-base)
(printout t "Below are the cases recalled from the case base: " t )
(for ?case cbr:in-cases-of ?case-base collect$ ?case )        )

(define-function recall-back ()
(printout t "1. Back to main: " t)
        (bind ?m (read))
                (if (= ?m 1) then (enter)
        else
(printout t "Thank You for using this system!:" t ))
```

Figure 5.4: Cases Display Function

```
(define-function recall-attribute-value (?case ?attribute )
(for ?value cbr:in-case-attribute-values-of herb-case-base ?case ?attribute collect$
?value) )
(define-function preprint-details ( )
(printout t "Please input case: ")
(bind ?m (read))
(print-details ?m)
        (printout t " "t t)
        (printout t "1. Display details of another case: " t)
        (printout t "2. Back to main: " t)
        (bind ?m (read))
        (if (= ?m 1) then (preprint-details)
        else
                (if (= ?m 2) then (enter)
        else
(printout t "Thank You for using this system!: " t))) )
(define-function print-details (?case)
    (printout t "Common Name : " (recall-attribute-value ?case common_name) )
    (printout t " (" (recall-attribute-value ?case chinese_name) ") "t)
    (printout t "Main Function: " (recall-attribute-value ?case main_function) t)
    (printout t "Sub Function: " (recall-attribute-value ?case sub_function) t)
    (printout t "Cuation: " (recall-attribute-value ?case caution) t)
    (printout t "Toxicity: " (recall-attribute-value ?case toxicity) t)
    (printout t "Dosage: " (recall-attribute-value ?case dosage) t)).
```

Figure 5.5: Case Details Display Function

```
(define-function add-case (?class)
(for ?c in-instances-of ?class do
    (cbr:add-case herb-case-base ?c :ignore-undefined-attributes? t))     )

(define-function preadd-case ( )
(printout t "Please input Class name: ")
(bind ?m (read))
(add-case ?m)

        (printout t "1. Add another case to Case Base: " t)
        (printout t "2. Back to main:  " t)
        (bind ?m (read))

        (if (= ?m 1) then  (preadd-case)
        else
                (if (= ?m 2) then  (enter)
        else
        (printout t "Thank You for using this system!:  " t))))
```

Figure 5.6: New Cases Adding Function

5.3 ART SCRIPT IMPLEMENTATION

```
(rep:load "case-based-reasoning")
(rep:add-subsystem "untitled" "case-based-reasoning")

(define-instance herb-case-base cbr:case-base
(cbr:scoring-function cbr:score-with-stored-max))
```

Figure 5.7: Define Case Base

'case-based-reasoning' system must be loaded before case base is created by typing (rep:load "case-based-reasoning") in the Command Interpreter window. In this project, the default system, 'untitled' is being used. Therefore, it is necessary to make case-base-reasoning a subsystem of untitled in order to access to CBR functionality by typing (rep:add-subsystem "untitled" "case-based-reasoning") in the Command Interpreter window.

A case base is an instance of the class cbr:case-base and the scoring function will be explain later in this chapter.

```
(define-attribute category slot)
(cbr:define-attribute herb-case-base category cbr:word
    :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)
(define-attribute common_name slot)
(cbr:define-attribute herb-case-base common_name cbr:word
    :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)
(define-attribute chinese_name slot)
(cbr:define-attribute herb-case-base chinese_name cbr:word
    :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)
(define-attribute main_function slot)
(cbr:define-attribute herb-case-base main_function cbr:word
    :match-contribution 100 :mismatch-penalty 0 :absence-penalty 30)
(define-attribute sub_function slot)
(cbr:define-attribute herb-case-base sub_function cbr:word
    :match-contribution 40 :mismatch-penalty 40 :absence-penalty 10)
(define-attribute caution slot)
(cbr:define-attribute herb-case-base caution cbr:word
    :match-contribution -90 :mismatch-penalty 0 :absence-penalty -50)
(define-attribute toxicity slot)
(cbr:define-attribute herb-case-base toxicity cbr:word
    :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)
(define-attribute dosage slot)
(cbr:define-attribute herb-case-base dosage cbr:word
    :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)
```

Figure 5.8: Define Attributes of Case

Case features or attributes are stored in ART attributes. Each attribute has a type which determines characteristics of the attribute. In addition to its type, each attribute has parameters that are used in scoring such as the match contribution, the mismatch penalty, and the absence penalty.

The following predefined attribute types are being used in this project:

- cbr:string - uses exact string matching.
- cbr:symbol - converts symbols to strings and then uses exact string matching.

- cbr:word - follows text preprocessing to remove misspellings, ignored words, and suffixes. Attribute values from stored cases are recorded as words to match against.

When cases are defined as ART objects, object attributes and case attributes are defined as well. The object attribute definition defines the storage properties of an attribute such as whether the attribute accepts multiple values. The case attribute definition defines the matching characteristics of an attribute. If object attributes are defined but case attributes are not defined, the object attributes will be ignored by CBR.

```
(define-class herb bc:core
(category)
(common_name)
(main_function)
(sub_function)
(caution)
(toxicity)
(dosage))

(define-instance Anemarrhenae herb
(category "clearing internal heat" )
(common_name "N/L")
(main_function "purge fire")
(sub_function "clear heat, excess lung heat, Drains Heat from the Lower Burner")
(caution "spleen deficiency diarrhea, respiratory arrest , large drops in blood pressure" )
(toxicity "Injections of large doses of solutions of this herb have resulted in respiratory
arrest and large drops in blood pressure. However, the implications of this
observation for decoctions taken by mouth are unclear.")
(dosage "2-4 qian"))

(for ?c in-instances-of herb do
    (cbr:add-case herb-case-base ?c :ignore-undefined-attributes? t))
```

Figure 5.9: Define class and instance cases

A class, Herb is defined for the cases in a case base. Each case is an instance of this class. One of the reasons is to iterate over all of the ART objects representing cases as instances of the class.

The case are added to the case base by iterate over the instances of case class, Herb, calling cbr:add-case on each instance.

```
(for ?c in-instances-of Herb do
(cbr:add-case herb-case-base ?c :ignore-undefined-attributes? t))
```

The :ignore-undefined-attributes? keyword is set to t so that cbr:add-case would not generate an error for the attributes title and author.

```
(set-attribute-value herb-case-base cbr:index-file "case-base/herb.cbi")
(cbr:save herb-case-base)
```

Figure 5.10: Save Case Base

The case base, herb-case-base is saved to a file and, at a later time, restored from the file by calling cbr:index-file attribute of the case base.

```
(set-attribute-value herb-case-base cbr:index-file "case-base/herb.cbi")
(cbr:save herb-case-base)
```

Figure 5.11: Restore Case Base

To restore the case base, herb-case-base, an instance of cbr:case-base is defined with the cbr:index-file attribute set to the appropriate file name.

5.4 WORD PREPROCESSING IMPLEMENTATION

Word preprocessing is invoke by defining an attribute to be of type cbr:word. A spell checker serves as the basis for word preprocessing in CBR. The spell checker makes use of a number of lexicons to perform spelling correction, removal of ignored words, synonym replacement, and suffix removal (stemming).

5.5.1 Lexicons of Spell Checker

Central to the function of the spell checker within CBR are a number of lexicons.
The cbr:word attribute type provides for 10 lexicons. Each lexicon serves a different purpose during the preprocessing operation.

There are two types of lexicons in the spell checker:

- Ignore: These lexicons are lists of words. When a word from a text block is found in an Ignore lexicon, it is ignored; otherwise, it is tested against the remaining lexicons.
- Change: These lexicons are lists of pairs of words. When a word from a text block matches the first word in a pair, it is replaced by the second word in the pair.

Ignore lexicons simply prevent a set of words from being processed by other lexicons while Change lexicons substitute one word for another.

These two lexicon types provide a mechanism for leaving words in the text block or replacing those words with other words, but they do not provide any explicit mechanism for removing words from the text block. Word removal is accomplished with a Change lexicon by changing the word to a special string which is later removed by the preprocessor. Below are the purposes served by the 10 lexicons used with the cbr:word attribute type:

- Main lexicon
- Main Compressed lexicon
- Main Autocorrect lexicon
- Autocorrect lexicon
- Ignored Words lexicon
- Synonym lexicon
- Abbreviation Suffix lexicon
- Exception Suffix lexicon
- User lexicon
- Casebase lexicon

5.1.1 Word Preprocessing

The word preprocessing performed by the cbr:word attribute type is only performed if there is a Casebase lexicon. This lexicon will record words that have not been found in

any of the other lexicons. Word preprocessing iterates over each word in a block of text and, for each word through the following steps:

- The word is converted to lowercase.
- If the cbr:hex-numbers attribute of the cbr:word attribute type is set to :LEADING-0X or :ANY-HEX, then an attempt is made to process the word as a hex number. If cbr:hex-numbers is set to :LEADING-0X, then the hex number can have a leading "0x" or "0X"; otherwise, it cannot. If cbr:hex-numbers is set to :NO-HEX, then an attempt is made to process the word as a non-hex number.
- If the word is found to be a hex or non-hex number, then it is processed according to the setting of the cbr:number-handling attribute of the cbr:word attribute type. If cbr:number- handling is :DROP, the number is removed and preprocessing continues with the next word. If cbr:number-handling is :ADD-DIRECTLY, the number is added and reprocessing continues. If cbr:number-handling is :PREPROCESS, then preprocessing continues as though the word were not a number.
- The word is then checked for nonalphabetic characters. If it contains nonalphabetic characters, then it is processed according to the setting of the cbr:nonalpha-handling attribute of the cbr:word attribute type. If cbr:non-alphahandling is :DROP, the word is removed and preprocessing continues with the next word. If cbr:non-alpha-handling is :ADD-DIRECTLY, the word is added and preprocessing continues. If cbr:non-alpha-handling is :PREPROCESS,

then preprocessing continues as though the word contained all alphabetic characters.

- The word is checked against the lexicons designated in the cbr:word attribute type. If this check indicates that the word is misspelled, the word is stemmed (ie, suffixes are removed) and stemmed word is checked against the lexicons.
- If either the original or stemmed words are found in a Change type lexicon, then the word is replaced by the correction unless the correction is the string "@#$%" in which case the word is removed and preprocessing continues.
- If the stemmed word is considered misspelled, then it is processed for suggestions. Suggestion processing results in a list of suggested alternatives for the word along with scores for each alternative indicating how closely the alternative matches the original word.

5.5 AHP WEIGHTED k-NN ALGORITHM IMPLEMENTATION

From the research that I have done, I have discovered that many CBR algorithms are derivatives of the k-nearest neighbour (k-NN) method, which has a similarity function to generate classification from the stored cases. One of the drawbacks of using k-NN method is its performance which is sensitive to the definition of its similarity function. This will affect the accuracy of the case matching. Various distance functions with feature weights has been used to reduce the sensitivity of k-NN method.

In this project, I have chosen to use the concept of analytic hierarchy process (AHP)-weighted k-NN algorithm for my CBR system. The AHP weighted k-NN algorithm has been shown to achieve classification accuracy higher than pure k-NN algorithm. Besides, AHP methodology has been used in this CBR system to assign relative importance in case indexing and retrieving.

5.6.1 Traditional k-nearest neighbor algorithm

NN matching function has been widely used in CBR as indexing and retrieval methods. The NN matching function is a non-parametric classification algorithm based on assumptions of the independence of attributes in previous cases and the availability of rules and procedures for matching. The NN techniques provide a measure of how similar a previous case is to given problem.

From research, I have notice that the traditional NN function is sensitive to the presence of irrelevant features in the case representation due to its similarity function, the Euclidean distance function, assumes that all features are equally relevant and has equal impact on similarity computations.

The best solution for this weakness is to set the weights in the similarity function appropriately. It means more important attributes should be assigned larger weights than less important attributes while totally irrelevant attributes should be assigned zero weight. This will bring to the improvement of k-NN algorithm performance.

Some researchers have suggested that the weight of all features be acquired by domain knowledge from experts (kolodner, 1993). AHP methodology has been adopted for domain knowledge based features weighting. The domain knowledge is important and essential to the reasoning process. In this point, the AHP approach is useful and systematic technique for acquiring feature weights from domain experts.

5.5.2 AHP approach to k-NN algorithm

The AHP approach is a multi-criteria decision making method that uses hierarchic or network structures to represent a decision problem and then develops priorities for the alternatives based on the decision maker's judgements throughout the system. It address the issues of how to structure a complex decision problem, identify its criteria, measure

the interaction among them and finally synthesize all the information to arrive at priorities, which depict preferences.

The AHP will enable the decision makers to structure a complex problem in the form of a hierarchy and evaluate a large number of quantitative and qualitative criteria in a systematic manner using multiple criteria. It will attempt to resolve conflicts and analyze judgments through a process of determining the relative importance of set of activities or criteria.

5.5.3 Advantage of AHP approach

- Easy to use
- Allows for rapid replanning
- Can incorporate qualitative and subjective factors
- Uses a psychometrics scale to quantify managerial judgements
- Provides a methodology to measure the consistency of these judgement

5.5.4 AHP Modeling for Traditional Chinese Herb recommendation

The AHP modeling for Traditional Chinese Herbs recommendation is a new analogical framework, which integrates knowledge-guided and inductive retrieval methods for

weighted k-NN methodology. This integration method represents suitable case retrieval and indexing, and incorporates domain knowledge for features weighting.

AHP is a proper means for feature weighting and allows for the inclusion of domain knowledge in Traditional Chinese Herbs recommendation.

The AHP model of this project arrays herbs at three hierarchy levels. The model has two main categories, positive matching and negative matching. Positive Matching is assessing with Main Function and Sub Function while Negative Matching is assess with caution and toxicity. The weights priority of each feature is assign using the sum value of match contribution, mismatch penalty and absence. The value of match contribution, mismatch penalty and absence is different for every feature and is set by the user with the domain knowledge from experts.

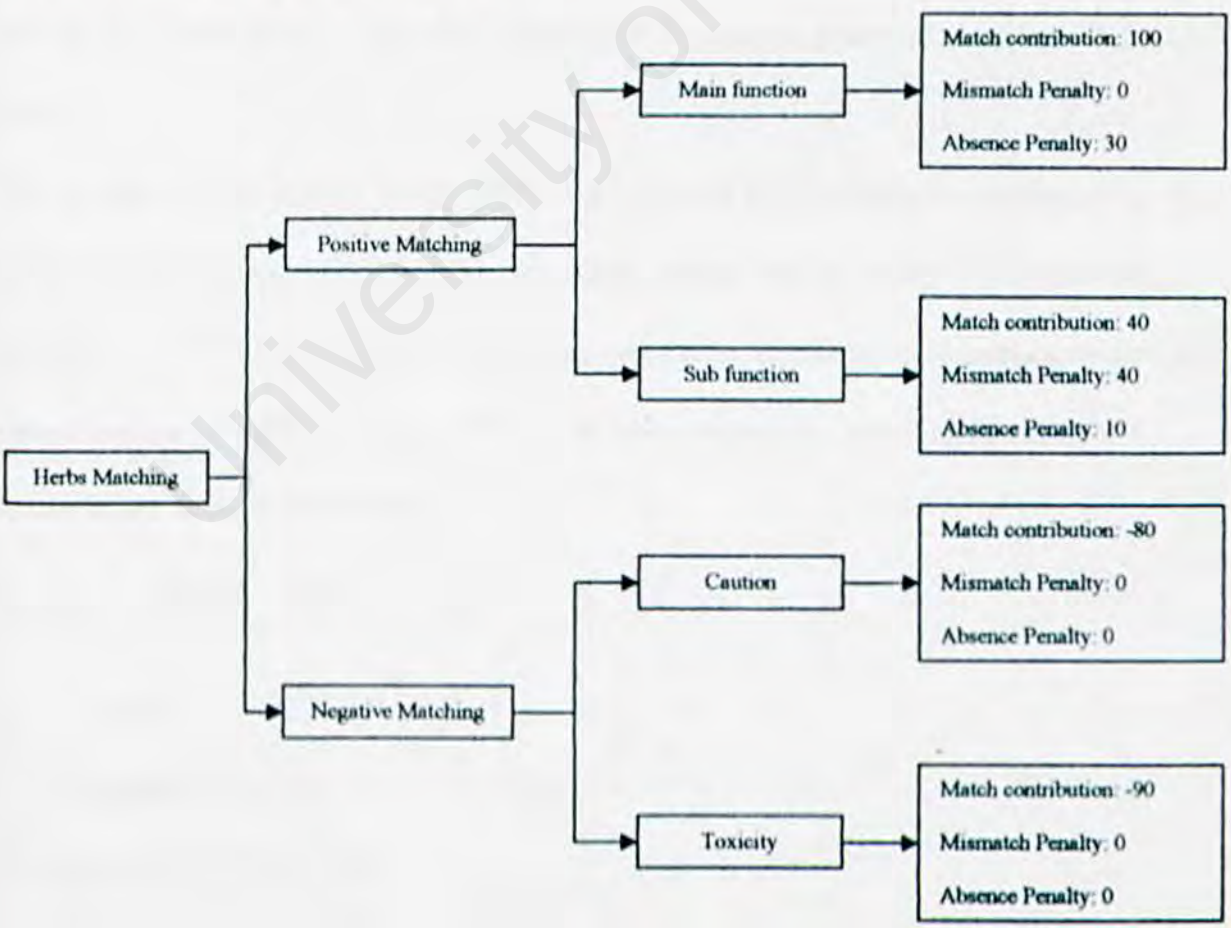


Figure 5.12: Feature weight of AHP model for Traditional Chinese Herbs

5.5.5 Indexing with AHP feature weight and weight value assign

In order to assign importance values to the features of cases is to assign weighting based on the knowledge of human experts. In this project, I have manage to ask for advices from an expert in Traditional Chinese Herb and gather information in order to decide which dimensions will bring to most accurate matching. In knowledge-guide indexing, domain knowledge is used to index cases directly. The hybrid indexing approach has significantly improved the speed and accuracy of case retrieval by storing the herb cases according to different categories.

The AHP model is effective in obtaining domain expert knowledge and representing knowledge-guided indexing. In this project, an integrated approach using AHP to assign knowledge based relative importance and CBR to retrieve more relevant case is being used.

The weighted k-NN model using AHP weight is used as an alternative methodology to assign the importance of features for case-based retrieving. By using this methodology, the features with weights of zero will be ignored during similarity computation while the features whose weights are high will have the most impact on determining similarity.

Below is the evaluation function:

$$\text{Dis}(x,y) = 1/n\sqrt{\sum_{1}^{n} w(x_i - y_i)^2}$$

1 if $x_i = y_i$

0 if $x_i \neq y_i$

for a given I (I = 1,2,.....,n)

n is the number of attributes, w is the eigen-vector matrix as the importance weight of the test data set on ith attribute, and the similarity of the values of the ith attributes of case x_i and y_i is sim $((x_i), (y_i))$ [0,1], the values for x_i and y_i in the input and retrieved cases.

5.5.6 Case retrieving with Inductive Retrieval algorithm

Since, case retrieval is a process that a retrieval algorithm retrieves the most similar cases to the current problem. Case retrieval requires a combination of search and matching.

Inductive retrieval is another widely applied algorithm in CBR applications and tools instead of k-NN algorithm. It is a good choice using nearest-neighbor retrieval without any preindexing [Watson, 1997]. If retrieval time becomes an important issue, inductive retrieval is preferable. Currently I am using AHP weighted k-NN algorithm in this project but the retrieval time will be slow when the case base is getting larger.

Inductive Retrieval technique has been implementing to solve this problem which means both inductive and AHP weighted k-NN techniques has been used. Inductive indexing is used to retrieve a set of matching cases, then AHP weighted k-NN is used to rank the cases in the set according to the similarity to the target case.

Table 5.1: Comparison between Nearest Neighbor Retrieval, AHP weighted k-NN retrieval and Inductive Retrieval

Retrieval Technique	Strength	Weakness
Nearest Neighbor Retrieval	Simple	Slow retrieval speed when the case base is large
AHP weighted k-NN Retrieval	Higher accuracy	Slow retrieval speed when the case base is large
Inductive Retrieval	Fast retrieval speed	Depends on pre-indexing which is a time-consuming process Impossible to retrieval a case while case data is missing or unknown

Inductive retrieval algorithm is a technique that determines which features do the best job in discriminating cases and generates a decision tree type structure to organize the cases in memory [Watson, 1997]. This approach is very useful when a single case's feature is required as a solution, and when that case feature is dependent upon others. Below is a decision tree generated from the herb cases in case base.

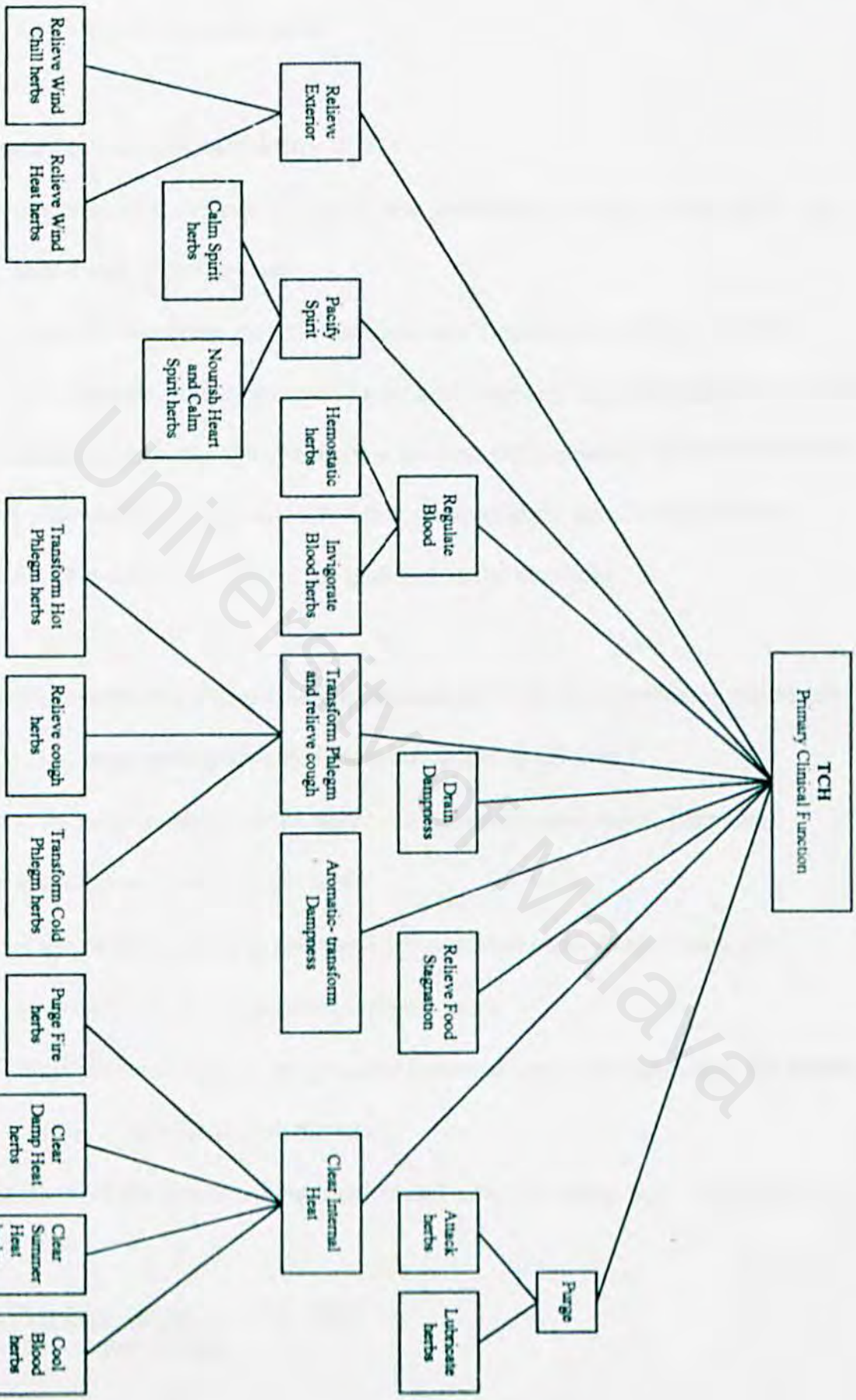


Figure 5.13: The decision tree of Traditional Chinese Herb

5.5.7 Similarity of Case Matching

The process of similarity calculation:

- For each attribute in the presented case, an attribute score is computed for each stored case in the case base
- For each stored case, the attribute score are combined to produce a case score
- The highest scores are returned by the match process. A threshold and maximum number of matches can be specified for a match. The number of scores returned by the match process will be less than or equal to the maximum number of matches and each score will be greater than the threshold.

The attribute score for a stored case is computed from a match contribution, mismatch penalty, or absence penalty specified for the attributes as follows:

- If the attribute values in the stored and presented cases match, the match contribution is added to the score.
- If the attribute values in the stored and presented cases do not match, the mismatch penalty is subtracted from the score
- If the attribute exists in the presented cases but not in the stored case, the absence penalty is subtracted from the score

The similarity of the case matching is calculated using cbr: score-with – stored-max:

$$\frac{\text{Attribute – score}}{\text{Maximum – stored – score}} \times 100$$

Scoring with cbr:score-with – stored-max will find the most complete matches for the presented case. It assumes that the presented case is complete otherwise, case with a great deal of data which perfectly match the presented case, will match poorly.

Limiting the number of Matches

The number of matches has been limited by setting two attributes of the case base:

1. **Cbr:match-threshold**

   To specifies a threshold that a case scores must succeed in order for the case to be included in the match set. Value 0 has been set for the system match threshold.

2. **Cbr:max-matches**

   To specifies the maximum number of matches in the match set. 10 matches have been set for the system maximum match.

5.5.8 Comparison between k-NN algorithm and AHP weighted k-NN algorithm

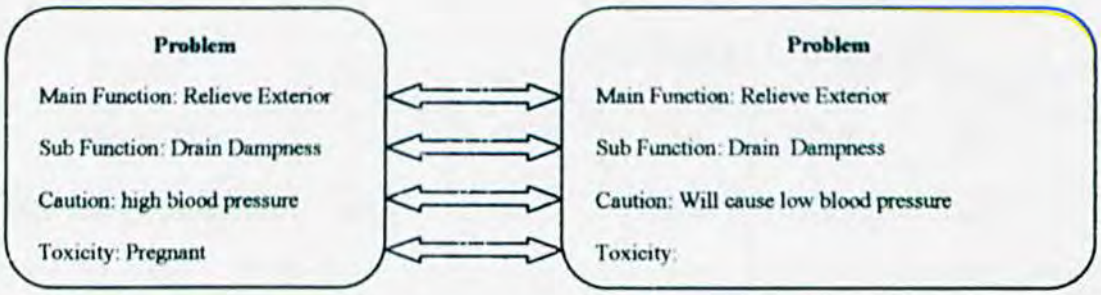


Figure 5.14: Overall similarity of Pure k-NN matching

Knowledge Nearest Neighbour (K-NN) function is sensitive to the presence of irrelevant features in the case representation (sub function, toxicity). Its similarity function, Euclidean distance function, assumes that all features have equal impact on similarity computation. As a result, the accuracy of the CBR matching will be low and unreliable.

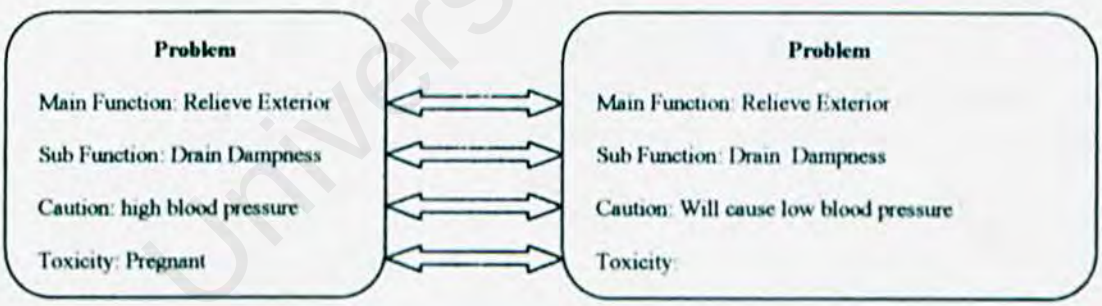


Figure 5.15: Overall similarity of AHP weighted K-NN model matching

By using the AHP weighted K-NN model, each feature will be assigned with different weight value according to different priority. Setting the weights in the similarity function appropriately can improve the performance of the traditional K-NN algorithms.

Table 5.2: Comparison Results of classification accuracy (Cheol-Soo Par*, Ingoo Han) between pure k-NN and AHP weighted k-NN

Model	Classification accuracy (%)					
	1st fold	2nd fold	3rd fold	4th fold	5th fold	Average
Pure k-NN	77.1	68.4	76.2	73.6	75.1	74.08
AHP weighted k-NN	83.1	84.2	84.4	86.3	84.7	84.52

From the table, the result of AHP weighted k-NN is probably better than pure k-NN with the average accuracy of 84.52% while pure k-NN is 74.08%.

CHAPTER 6: SYSTEM TESTING

6.1 INTRODUCTION

Testing is important in developing a system. The process of testing and debugging are for the purpose of detecting defects and bugs of a system. These processes are usually done incrementally with system development.

This phase of system testing is sometimes referred as Verification and Validation. Verification means a set of activities that ensure the system correctly implements a specific function while Validation means a different set of activities that ensure the system has been built is traceable to user requirements. A successful test is one in which no errors are found.

The following are the objectives for system testing:

- To reveal inference error by the case-based matching and case retrieval.
- To compare the expected outcome with the actual outcome. Eventually, debug it to enhance it functionality and capability.
- To ensure the accuracy of the case matching and make sure every case from the case can be retrieved for matching..

6.2 MODULE TESTING

Module testing focuses verification effort on the smallest unit of system design that is the system component or module.

Each module is tested independently to ensure its workability and error free. The interface module is tested for its ease of use, simplicity and ambiguity, and the system is tested on its knowledge acquisition capabilities efficiency and usability.

6.2.1 Case Matching Accuracy Testing

The accuracy of the case matching is tested by presenting a few known cases for matching. If the score for the matched case is exactly what has been expected, then there should be no problem with the accuracy. If not, then the problem would have either occurred from weighting value or retrieval algorithm or indexing. The decision tree or the herbs modeling might need to rearrange. If the problems occurred from weight value, then the weight value will be reassigned and weight testing is necessary.

6.2.2 Weighting Module Testing

In this project, the system is based on AHP weighted K- Nearest Neighbor algorithm, the weight is in numerical form. Every attribute of a case is assign with different weight value. The herbs are being modeled into a three level hierarchy with two main

categories, positive matching and negative matching. The weights priority of each attribute is assign using the sum value of match contribution, mismatch penalty and absence. The value of match contribution, mismatch penalty and absence is assigned to this system and by using a 'score-with-stored-max' calculation function, the weight value is then assigned to the attribute.

A number of known cases are presented to the system with different weight value for every attribute in order to define the accuracy. A statistic is made to understand the effect of different weight value to the matching accuracy. The set of weight values that suit the system the most will be selected as the weight value for the case attributes.

6.2.3 Case Base Module Testing

This is referring to the case base testing to ensure that every data which user inserted from the Art*Enterprise command interpreter is stored accurately and correctly to the correspondence case base which is in the *.cbi format. A set of sample raw data is created for the testing purpose in this module. The process includes iterate the checking on the duplicated data in database to ensure that every entered data is valid and ease the redundancy and duplication problem.

6.2.4 Coding Module Testing

Times of testing have been done on the AHP weighted K-NN algorithm to ensure case retrieval and case matching with highest accuracy and response time.

In this project, the comparison results between AHP weighted K-NN model and pure k-NN are showed to reveal an important improvement in case matching accuracy and faster response time for a large case-base by this implementation.

A set of Art Script coding is prepared to assign the importance of weights to each feature for knowledge-guide retrieval and indexing.

```
(define-attribute main_function slot)
(cbr:define-attribute herb-case-base main_function cbr:word
    :match-contribution 100 :mismatch-penalty 0 :absence-penalty 30)
(define-attribute sub_function slot)
(cbr:define-attribute herb-case-base sub_function cbr:word
    :match-contribution 40 :mismatch-penalty 40 :absence-penalty 10)
(define-attribute caution slot)
(cbr:define-attribute herb-case-base caution cbr:word
    :match-contribution -90 :mismatch-penalty 0 :absence-penalty -50)
(define-attribute toxicity slot)
(cbr:define-attribute herb-case-base toxicity cbr:word
    :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)
```

Figure 6.1: Samples of Art Script for Case Base reasoning.

To test whether the case retrieval process is working or not, I must make sure that each of the attribute of cases have been assign a specific weights with different priorities. The K-NN model with AHP weights and Inductive Retrieval model has been implemented to create a case-based retrieval algorithm. This methodology is used to assigned relative

importance of feature weighting for case-indexing and fast response time for case retrieval.

6.3 INTEGRATION TESTING

In this project, a bottom-up approach has been used for system testing. It begins construction and testing with modules at the lowest levels of the system and then moving upward to the modules at the higher levels of the system. Since all the modules have been tested and have been declared bug free, the modules will be combined. The Integration is checked again to ensure there is no error. The matching and retrieving process are tested thoroughly to demonstrate the essence of case base methodology.

6.4 SYSTEM TESTING

System testing is a series of different tests designed to fully exercise the system to uncover its limitations and measure its capabilities. All modules are combined to complete the system and it is tested as a whole to ensure workability and error free.

The objective is to test the whole system and verify that it meets specified requirements. Testing is concentrated on the whole reasoning process from past case acquisition until evolution in system and retainment of solved case. There are several types of system testing that are worthwhile for a system.

6.4.1 Rule Testing

It is a test during the run time environment where the complete set of rules and functions are loaded. If the Art*Enterprise can run the rules and functions of smoothly without error, then it has considered success.

6.4.2 Performance Testing

The purpose of this testing is to test the run-time performance of the system in case retrieving for matching. Different types of indexing have been used to calculate the performance of case retrieval and the one with fastest performance has been chosen for the case indexing of this system.

6.5 TESTING ANALYSIS

The system testing shows that the system able to display the expected match cases for either complete or uncompleted presented case, the weight scores is being display as well for user references. Besides, the system is easy to use and understand. As a conclusion, all the objectives have been achieved.

CHAPTER 7: SYSTEM EVALUATION & CONCLUSION

7.1 SYSTEM EVALUATION

Various problems were encountered when developing this system. These problems were solved through research and studies in fields such as the Internet, Bulletin Board Website, journals and reference book. The system's strengths, limitations, and future enhancement were identified.

7.2 PROBLEMS ENCOUNTERED AND SOLUTIONS

Problems are everywhere and so does in every system. Several problems encountered throughout the development of this system. These include:

7.2.1 Difficulty in choosing development technology and tools

There are many software tools available to develop case-based reasoning system. Choosing a suitable technology and tools was a critical process as all tools possesses their own strengths and weaknesses. In addition, the availability of the required tools for development was also a major consideration. Eventually, Art*Enterprise has been chosen of its strength in case-based reasoning time and recommendation by my supervisor of this project, Assoc. Prof. Dr. Syed Malek Fakar Duani .

7.2.2 Lack of knowledge of Traditional Chinese Herbs

Traditional Chinese Medicine has been a totally new subject for me to learn and understand in order to create a good case representation for this system. As noted, successful performance of retrieval mechanism depends very much on good representation too beside indexing and similarity metric.

Books and articles from internet have been my main resource in understanding the concept of Traditional Chinese Herbs. Besides, I have gathered some information from my relative who is currently working in a Traditional Chinese Herbs Pharmacies.

7.2.3 Lack of knowledge Case-Based Reasoning

Nothing is too difficult in terms of theoretical. All the problems and doubts will emerge once the development and design starts. It's hard to create a good case representation, hard to choose an appropriate algorithm for case retrieval and indexing and the type of similarity should be used for case matching as well.

I have managed to read many articles and journal for references and many resources have been found through internet.

7.2.4 Lack of Knowledge in choosing and implement an appropriate algorithm for case retrieval and indexing

The very common case retrieval and indexing algorithm, K-NN algorithm has been implemented into my system but it is sensitive to the presence of irrelevant features in the case representation and the speed of case retrieving will become slow when the case

base is getting larger. The solution is to assign weight priorities on different features in the case representation. But, they are quite a few researches have been empirical work on the weight setting on K-NN algorithm. It is a difficult job to find out the moist suitable algorithm for my system and to understand the algorithms as well by reading the journals and articles. I have chosen AHP weighted K-NN algorithm and for a better improvement of case retrieval and indexing and implemented Inductive Retrieval Algorithm for faster case retrieval.

7.2.5 Lack of Knowledge in Art Script

Since there was no prior knowledge of programming in Art Script, there was an uncertainty on how to write and organize the codes. These new programming languages and concepts were never taught before and it takes time to understanding the programming Art * Enterprise as well in order to use it for writing Art Script and compilation. Fortunately, previous knowledge in traditional object-oriented programming language like programming C++ did help me in understanding the Art Script since the Art Script is based on object-oriented too.

7.3 SYSTEM STRENGTHS

During the development of this project, several system strengths were identified and described as follow:

- The system is capable of capturing knowledge in a heuristic manner unlike structured and restricted knowledge in the form of if then clauses. Knowledge of this system is based on past case as a whole. Hence, this will capture the entire knowledge content not only specific detail knowledge.
- This system can be a good trainer and can assist inexperienced Traditional Chinese Medical trainees or even professional Traditional Chinese Medical Doctor in deciding combination of herbs for certain diseases. It can also be used as a learning tool to train new student and minimize the risk of poor decision making by these young and inexperienced doctors.
- The system is provided with a spell checker that serves as the basis for word processing in CBR. The spell checker makes use of a number of lexicons to perform spelling correction, removal of ignored words, synonym replacement and suffix removal.
- The system has the capability of knowledge acquisition on past cases and new herbs. This system has being designed to allow user to enter past cases into the case base.
- By the implementation of CBR, this system is very consistent in decision making and unlike human expert; its intelligence will increase overtime as the case in the

case base increases. Therefore, this system is dynamic and able to keep with the fast changing knowledge and situation.

7.4 SYSTEM LIMITATIONS

Due to the time constraint and the programming language itself, there were some limitations in this system. These include:

- Due to the software constraint of Art * Enterprise, no Graphical User Interface (GUI) has been implement to this system. The user will need to type in the commands into the command interpreter in order to run this system.

- The scope is limited and restricted. Hence, it does not result the full treatment for Traditional Chinese Herbs medical but it's a good start up project for the industry.

Functions have been added to the system in order to make the system more easy to use by just typing in the functions and the parameters needed.

7.5 PROBLEM IN IMPLEMENTATION

Art* Enterprise does not allow the creation of executable file (*.exe). Hence, the users must have minimum knowledge on art script commands before they can operate the system.

Lacking of documentation in Art Enterprise is also a major problem although vast amount of references and documentation is available on CBR concept and implementation. The Brightware, inc. has currently stop launching the Art * Enterprise. The information that I can get about the software is very little and very few references have been found.

7.5 FUTURE ENHANCEMENT

Further development and many new ideas have come about while the system was being implemented but owing to time constraint and other factors, not all of the ideas could be incorporated into the system. For example, the efficiency in data entry of new cases, the restricted amount of relevant question asked by system, and last but not least the scope of the project which is so confined and restricted that it does not reflect the full requirement of the Traditional Chinese Herbs Medical Treatment.

The following aspects should be considered in future enhancement:

a) Efficiency of data entry of new case and case matching

The system should need some improvement in system user friendliness. More information should be gathering for a case matching which will result in more features of every case. So far, the inefficiency in data entry of new cases and the restricted amount of relevant question asked by system should receive priority in future work developments.

b) Enlarge the scope of the project

In order to make the system more complete and commercialize, more expert should be participated in this project to enlarge the scope of the project.

c) Graphic User Interface

A fully developed and complete system should have a good Graphic User Interface (GUI) so that it is easy for the end user to use and understand. A system is considered no commercial value if nobody can use the system or a vast amount of time is needed to understand it.

The absence of industry and expert participation is currently one of the constraints faced by this project. However, CBR has been proved to be an alternative in Traditional Chinese Herbs Medical Treatment through this project.

BIBLIOGRAPHY

[1] Louis E. Frenzel, "Crash Course in Artificial Intelligence and Expert System", Howard W. Sams & Co., 1987.

[2] Mohamed H Hassaun, "Fundamental of Artificial Neural Network", Bradford Book, 1995.

[3] Shari Lawrence Pfleeger, "Software Engineering - Theory and Practice", Prentice Hall International Inc, 1998.

[4] David B. Leake (1996). "Case-Based Reasoning: Experiences, Lessons and Future Directions", Menlo Park: AAAI Press/ MIT Press, 1996.

[5] Whitten, Jeffrey L. & Bently, Lonnie D., "System Analysis and Design Methods", McGraw-Hill International Editions, 1998.

[6] Janet Kolodner, "Case-Basewd Reasoning", Morgon Kaufmann Publisher, 1993

[7] http://www.ceng.metu.edu.tr/~e126436/Ceng568/CBR-paper.htm, Date Referred: June 2002

[8] http://www.ecfc.u-net.com/cost/case.htm, Date Referred: June 2002

[9] www.ai-cbr.org, Date Referred: June 2002

[10] http://www.cbr-web.org, Date Referred: June 2002

[12] http://online.loyno.edu, Date Referred: June 2002

[13] http://www.aiai.ed.ac.uk/links/cbr.html, Date Referred: June 2002

[14] http://www-cia.mty.itesm.mx/~lgarrido/Repositories/CBR/reasoning.html, Date Referred: June 2002

[15] http://www.botknowledge.com, Date Referred: June 2002

[16] http://www.howstuffwork.com, Date Referred: July 2002

[17] http:// www.Rnhiherbal.org, Date Referred: June 2002

[18] http://www.usatoday.com/life/cyber/tech/2001-06-20-ai-history.htm, Date Referred: June 2002

[19] http://www.wellspace.com, Date Referred: June 2002

[20] http://www.chineseherb.com, Date Referred: July 2002

[21] http://www.jungtao.edu, Date Referred: July 2002

[22] http://www.askdrwang.com, Date Referred: July 2002

[23] http://library.thinkquest.org, Date Referred: August 2002

[24] http://www.teklearning.com, Date Referred: August 2002

[25] http://www.microsoft.com, Date Referred: August 2002.

[26] Cheol-Soo Park*, Ingoo Han, "A Case-based reasoning with the feature weights derived by analytic hierarchy process for bankruptcy prediction", Graduate. School of Management, Korea advanced Institute of Science and Technology.

[27] Rainer Schmidt*, Lothar Gierl, "Case-based reasoning for antibiotics theraphy advice: an investigation of retrieval algorithms and prototypes", Institute for Medical Informatics and Biometry, University of Rostock, 2001.

[28] Dr. John Hunt, "Case based Diagnosis / classification", JayDee Technology Ltd.,1999.

[29] Ian Watson, "CBR is a Methodology *not* a Technology", AI-CBR, University of Salford, 1998.

```
********************************************************************
/* Load the cbr system and subsystem */
********************************************************************
(rep:load "case-based-reasoning")
(rep:add-subsystem "medicine" "case-based-reasoning")

********************************************************************
/* Create case base and define scoring function */
********************************************************************
 (define-instance herb-case-base cbr:case-base
        (cbr:index-file "case-base/herb.cbi")
 (cbr:scoring-function cbr:score-with-stored-max))

********************************************************************
/* Define case attributes or features*/
********************************************************************
 (define-attribute category slot)
(cbr:define-attribute herb-case-base category cbr:word
      :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)

(define-attribute common_name slot)
(cbr:define-attribute herb-case-base common_name cbr:word
      :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)

(define-attribute chinese_name slot)
(cbr:define-attribute herb-case-base chinese_name cbr:word
      :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)

(define-attribute main_function slot)
(cbr:define-attribute herb-case-base main_function cbr:word
      :match-contribution 100 :mismatch-penalty 0 :absence-penalty 30)

(define-attribute sub_function slot)
(cbr:define-attribute herb-case-base sub_function cbr:word
      :match-contribution 40 :mismatch-penalty 40 :absence-penalty 10)

(define-attribute caution slot)
(cbr:define-attribute herb-case-base caution cbr:word
      :match-contribution -90 :mismatch-penalty 0 :absence-penalty -50)

(define-attribute toxicity slot)
(cbr:define-attribute herb-case-base toxicity cbr:word
      :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)

(define-attribute dosage slot)
(cbr:define-attribute herb-case-base dosage cbr:word
 :match-contribution 0 :mismatch-penalty 0 :absence-penalty 0)
```

```
*************************************************************************
/* Define class */
*************************************************************************
(define-class herb bc:core
(category)
(common_name)
(main_function)
(sub_function)
(caution)
(toxicity)
(dosage))

*************************************************************************
/* The main display function for the system */
*************************************************************************
(define-function enter ()
(printout t "This is a CBR system for Traditional Chinese Herbs medical diagnosis: " t)
(printout t "Please choose from the following options for execution: " t)
(printout t "1. Display Cases from Case Base " t)
(printout t "2. Display Case details " t)
(printout t "3. Insert new case" t)
(printout t "4. Add new cases to case base " t)
(printout t "5. Search for combination of Traditional Chinese herbs for certain disease.
" t)
(printout t "6. Retain presented case to case base " t)

/* Read the input from the user*/
(bind ?m (read))
   (if (= ?m 1) then (recall-case herb-case-base)
                else
                (if (= ?m 2) then  (preprint-details)
                else
                (if (= ?m 3) then   (insert)
                else
                (if (= ?m 4) then  (preadd-case)
                else
                (if (= ?m 5) then  (diagnose)
                else
                (if (= ?m 6) then  (retain)
                else
                (printout  t "Wrong input! Please try again... " t ))))))))

*************************************************************************
/* The matching function */
*************************************************************************
(define-function check-nil (?m)
     (if(or(= ?m "NIL")
```

```
        (= ?m "nil"))    then
      (= ?m """)))

/* Define global variable*/
(define-global ?*main_function* = some-value)
(define-global ?*sub_function* = some-value)
(define-global ?*caution* = some-value)
(define-global ?*toxicity* = some-value)

/*define presented case and set the default  attribute value*/
(define-instance presented-case herb
   (main_function ?*main_function*)
   (sub_function ?*sub_function*)
   (caution ?*caution*)
   (toxicity ?*toxicity*))

/*Display for user for insert presented case*/
(define-function diagnose ()
        (printout t "Main Function? ")
        (bind ?*main_function* (read))
        (printout t "Sub Function? ")
        (bind ?*sub_function* (read))
        (printout t "Caution?")
        (bind ?*caution* (read))
        (printout t "Toxicity?")
        (bind ?*toxicity* (read))

/*set new value for case attributes*/
(set-attribute-values presented-case
                main_function ?*main_function*
                sub_function ?*sub_function*
                caution ?*caution*
                caution ?*toxicity*)

        /*validation for nil"
        (check-nil ?*main_function*)
        (check-nil ?*sub_function*)
        (check-nil ?*caution*)
        (check-nil ?*toxicity*)

        /*Display the options for the user*/
        (printout t "Choose the type of list for display from the following options: " t)
        (printout t "1. List for Doctor references " t)
        (printout t "2. List for Patient references " t)
        (bind ?m (read))

        (if (= ?m 1) then  (print-match herb-case-base presented-case)
```

```
        else
        (if (= ?m 2) then (print-list herb-case-base presented-case)))

        (printout t "1. Search for another matching: " t)
        (printout t "2. Back to main: " t)
        (printout t "3. End: " t)
        (bind ?m (read))

        (if (= ?m 1) then (diagnose)
        else
                (if (= ?m 2) then (enter)
        else
                (printout t " Thank you for using this system!" t))))

*************************************************************************
/* Print matched case with display properties for Doctor Reference */
*************************************************************************
(define-function presented-case (?presented)
(cbr:match-case herb-case-base ?presented)
(cbr:matches-found herb-case-base))


(define-function print-match (?case-base ?presented)
(presented-case ?presented)
(for ?m from 1 to 1 do
(printout t "Below are the Traditional Chinese Herbs recommended for "
(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-case ?case-
base ?m) category collect$ ?value) ": " t t))

(for ?m from 1 to (cbr:matches-found ?case-base) do
        (printout t "Match " ?m "   "      " Score: "
            (cbr:get-match-score ?case-base ?m)  t
"Herb: "(cbr:get-match-case ?case-base ?m) t
"Common Name: "(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-
match-case ?case-base ?m) common_name collect$ ?value) t

"Toxicity : "(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-
case ?case-base ?m) toxicity collect$ ?value)
    t t)))
(print-match herb-case-base presented-case-1)

*************************************************************************
/* Print matched case with display properties for patient reference */
*************************************************************************
(define-function print-list (?case-base ?presented)
(presented-case ?presented)
(for ?m from 1 to 1 do
```

```
(printout t "Below are the Traditional Chinese Herbs recommended for "
(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-case ?case-
base ?m) category collect$ ?value) ": " t t))

(for ?m from 1 to (cbr:matches-found ?case-base) do
        (printout t "("?m ")" (cbr:get-match-case ?case-base ?m) " "
(for ?value cbr:in-case-attribute-values-of ?case-base (cbr:get-match-case ?case-
base ?m) dosage collect$ ?value) t t)))
(print-list herb-case-base presented-case-1)

*************************************************************************
/* Display cases of case base */
*************************************************************************
(define-function recall-case (?case-base)
(printout t "Below are the cases recalled from the case base (restart after finish
loading): " t )
(for ?case cbr:in-cases-of ?case-base collect$ ?case )          )

(define-function recall-back ()
(printout t "1. Back to main: " t)

        (bind ?m (read))
                (if (= ?m 1) then (enter)
        else
(printout t "Thank You for using this system!:" t )))

*************************************************************************
/* Print case details */
*************************************************************************
 (define-function recall-attribute-value (?case ?attribute )
(for ?value cbr:in-case-attribute-values-of herb-case-base ?case ?attribute
collect$ ?value) )

(define-function preprint-details ( )
(printout t "Please input case: ")
(bind ?m (read))
(print-details ?m)
        (printout t " "t t)

        (printout t "1. Display details of another case: " t)
        (printout t "2. Back to main: " t)
        (printout t "3. End: " t)

        (bind ?m (read))

        (if (= ?m 1) then (preprint-details)
        else
```

```
            (if (= ?m 2) then  (enter)
        else
(printout t "Thank You for using this system!:  " t)) )
(define-function print-details (?case)
      (printout t "Common Name : " (recall-attribute-value ?case common_name) )
      (printout t "  (" (recall-attribute-value ?case chinese_name) ") "t)
      (printout t "Main Function: " (recall-attribute-value ?case main_function) t)
      (printout t "Sub Function: " (recall-attribute-value ?case sub_function) t)
      (printout t "Caution: " (recall-attribute-value ?case caution) t)
      (printout t "Toxicity: " (recall-attribute-value ?case toxicity) t)
      (printout t "Dosage: " (recall-attribute-value ?case dosage) t))

*************************************************************************
/* Add new case to case base */
*************************************************************************
 (define-function add-case (?case)
 (cbr:add-case herb-case-base ?case :ignore-undefined-attributes? t)    )

(define-function preadd-case ( )
(printout t "Please input case name: ")
(bind ?m (read))
(add-case ?m)

        (printout t "1. Add another case to Case Base: " t)
        (printout t "2. Back to main:  " t)
        (printout t "3. End:  " t)
        (bind ?m (read))
        (if (= ?m 1) then  (preadd-case)
        else
                (if (= ?m 2) then  (enter)
        else
        (printout t "Thank You for using this system!:  " t))))

*************************************************************************
/* Retain cases into the case base*/
*************************************************************************
(define-function retain ( )
(set-attribute-value herb-case-base cbr:index-file "case-base/herb.cbi")
(cbr:save herb-case-base)
(printout t "Retain successfully..... " t)
(enter))

*************************************************************************
/* Insert past case or herbs into the system */
*************************************************************************
(define-function insert ( )
(printout t "Please insert new case in required format (restart after finish loading):  " t))
```